L 5

```
   SSSSSSSSSSSS  YYY         YYY     SSSSSSSSSSSS  LLL                 OOOOOOOOO        AAAAAAAAA
   SSSSSSSSSSSS  YYY         YYY     SSSSSSSSSSSS  LLL                 OOOOOOOOO        AAAAAAAAA
   SSSSSSSSSSSS  YYY         YYY     SSSSSSSSSSSS  LLL                 OOOOOOOOO        AAAAAAAAA
SSS              YYY         YYY  SSS              LLL              OOO         OOO  AAA         AAA
SSS              YYY         YYY  SSS              LLL              OOO         OOO  AAA         AAA
SSS              YYY         YYY  SSS              LLL              OOO         OOO  AAA         AAA
SSS                 YYY   YYY     SSS              LLL              OOO         OOO  AAA         AAA
SSS                 YYY   YYY     SSS              LLL              OOO         OOO  AAA         AAA
SSS                 YYY   YYY     SSS              LLL              OOO         OOO  AAA         AAA
   SSSSSSSSS           YYY           SSSSSSSSS     LLL              OOO         OOO  AAA         AAA
   SSSSSSSSS           YYY           SSSSSSSSS     LLL              OOO         OOO  AAA         AAA
   SSSSSSSSS           YYY           SSSSSSSSS     LLL              OOO         OOO  AAA         AAA
            SSS        YYY                    SSS  LLL              OOO         OOO  AAAAAAAAAAAAAAA
            SSS        YYY                    SSS  LLL              OOO         OOO  AAAAAAAAAAAAAAA
            SSS        YYY                    SSS  LLL              OOO         OOO  AAAAAAAAAAAAAAA
            SSS        YYY                    SSS  LLL              OOO         OOO  AAA         AAA
            SSS        YYY                    SSS  LLL              OOO         OOO  AAA         AAA
            SSS        YYY                    SSS  LLL              OOO         OOO  AAA         AAA
SSSSSSSSSSSS           YYY        SSSSSSSSSSSS     LLLLLLLLLLLLLLL     OOOOOOOOO     AAA         AAA
SSSSSSSSSSSS           YYY        SSSSSSSSSSSS     LLLLLLLLLLLLLLL     OOOOOOOOO     AAA         AAA
SSSSSSSSSSSS           YYY        SSSSSSSSSSSS     LLLLLLLLLLLLLLL     OOOOOOOOO     AAA         AAA
```

```
_$2

Sym
---
SS1
SS1

SS1
SS1
SS1
SS1
SS1
SS1
SYS
SYS
SYS
TRY
UNL
WRI
```

```
**FILE**ID**INIADPUV1                                  I 14
```

```
IIIIII    NN      NN    IIIIII      AAAAAA    DDDDDDDD    PPPPPPPP    UU      UU  VV      VV      11
IIIIII    NN      NN    IIIIII      AAAAAA    DDDDDDDD    PPPPPPPP    UU      UU  VV      VV      11
  II      NN      NN      II      AA      AA  DD      DD  PP      PP  UU      UU  VV      VV    1111
  II      NN      NN      II      AA      AA  DD      DD  PP      PP  UU      UU  VV      VV    1111
  II      NNNN    NN      II      AA      AA  DD      DD  PP      PP  UU      UU  VV      VV      11
  II      NNNN    NN      II      AA      AA  DD      DD  PP      PP  UU      UU  VV      VV      11
  II      NN  NN  NN      II      AA      AA  DD      DD  PPPPPPPP    UU      UU  VV      VV      11
  II      NN  NN  NN      II      AA      AA  DD      DD  PPPPPPPP    UU      UU  VV      VV      11
  II      NN    NNNN      II      AAAAAAAAAA  DD      DD  PP          UU      UU  VV      VV      11
  II      NN    NNNN      II      AAAAAAAAAA  DD      DD  PP          UU      UU  VV      VV      11
  II      NN      NN      II      AA      AA  DD      DD  PP          UU      UU    VV  VV        11          ....
  II      NN      NN      II      AA      AA  DD      DD  PP          UU      UU    VV  VV        11          ....
IIIIII    NN      NN    IIIIII    AA      AA  DDDDDDDD    PP          UUUUUUUUUU      VV      111111        ....
IIIIII    NN      NN    IIIIII    AA      AA  DDDDDDDD    PP          UUUUUUUUUU      VV      111111        ....

LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

Table of contents

```
    (3)       254         Macros to describe nexus configurations
    (4)       379         Adapter-specific data structures
    (5)       523         CPU-specific data structures
    (6)       723         Message strings
    (7)       734         INI$IOMAP, Initialize and map nexuses
    (8)       899         INITADP_780, _750, _730, and _UV1
    (9)       916         CONFIG_IOSPACE
    (10)     1066         CREATE_ARRAYS
    (11)     1109         MAP_PAGES
    (13)     1269         INI$UBSPACE
    (14)     1339         INI$UBADP - BUILD ADP AND INITIALIZE UBA
    (14)     1815         INI$MBADP - BUILD ADP AND INITIALIZE MBA
    (14)     1816         INI$DRADP - BUILD ADP AND INITIALIZE DR32
    (14)     1817         INI$CIADP - BUILD ADP AND INITIALIZE CI
    (14)     1997         INI$KDZ11
    (14)     2031         INI$CONSOLE, init data structures for console
    (15)     2141         EXE$INI_TIMWAIT - COMPUTE CORRECT TIMEWAIT LOOP VALUES
    (16)     2299         EXE$INIT_TODR  -  SET SYSTEM TIME TO CORRECT VALUE AT STARTUP
```

```
0000          .NLIST  CND
0000
0000
0000
0000
0000          .TITLE  INIADPUV1 - ADAPTER INITIALIZATION FOR MICRO-VAX I
0000
0000
0000          .IDENT  'V04-002'
0000     ;
0000     ;****************************************************************************
0000     ;*                                                                          *
0000     ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
0000     ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
0000     ;*  ALL RIGHTS RESERVED.                                                    *
0000     ;*                                                                          *
0000     ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
0000     ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0000     ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
0000     ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
0000     ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
0000     ;*  TRANSFERRED.                                                            *
0000     ;*                                                                          *
0000     ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
0000     ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0000     ;*  CORPORATION.                                                            *
0000     ;*                                                                          *
0000     ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
0000     ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
0000     ;*                                                                          *
0000     ;*                                                                          *
0000     ;****************************************************************************
0000     ;
0000     ; Facility: System bootstrapping and initialization
0000     ;
0000     ; Abstract: This module contains initialization routines that are loaded
0000     ;           during system initialization (rather than linked into the system).
0000     ;
0000     ; Environment: Mode = KERNEL, Executing on INTERRUPT stack, IPL=31
0000     ;
0000     ; Author:  Trudy C. Matthews            Creation date: 22-Jan-1981
0000     ;
0000     ; Modification history:
0000     ;
0000     ;       V04-002 TCM0013         Trudy C. Matthews       10-Sep-1984
0000     ;               Add $BQODEF missing from TCM0012.
0000     ;
0000     ;       V04-001 TCM0012         Trudy C. Matthews       07-Sep-1984
0000     ;               For venus processor:  turn on cache before calibrating
0000     ;               TIMEDWAIT cells (routine EXE$INI_TIMWAIT).  Store the TIMEDWAIT
0000     ;               values calculated after cache is enabled in the boot driver's
0000     ;               TIMEDWAIT cells.  This is because the boot driver initially
0000     ;               has to run with cache off, but after booting will run with
0000     ;               cache on.
0000     ;
0000     ;       V03-024 TCM0011         Trudy C. Matthews       31-Jul-1984
0000     ;               Change venus's CRD interrupt vector back to ^X54 in the SCB,
0000   ;               and its SBIA Fail vector to ^X64.
0000   ;
0000   ;       V03-023 WMC0001          Wayne Cardoza           30-Jul-1984
0000   ;               Add H memory to 780 list.
0000   ;
0000   ;       V03-022 TCM0010          Trudy C. Matthews       25-Jul-1984
0000   ;               Fix a bug in INI$UBSPACE for the 11/790 that caused second
0000   ;               and subsequent unibus adapter spaces to be mapped incorrectly.
0000   ;               Fix bugs in INI$SCB for the 11/790.  Fix conditional
0000   ;               assembly flags in INI$CONSOLE for the 11/790.
0000   ;
0000   ;       V03-021 KDM0100          Kathleen D. Morse       01-May-1984
0000   ;               Correct address of memory CSRs to be past the 8 missing
0000   ;               Qbus adapter pages that do not exist.
0000   ;
0000   ;       V03-020 KDM0099          Kathleen D. Morse       27-Apr-1984
0000   ;               On a MicroVAX I, if the sysgen parameter TIMEDWAIT is set
0000   ;               to request no time-prompting, then use the last recorded
0000   ;               system time instead.  This is found in EXE$GQ_TODCBASE
0000   ;               which can be updated with a SET TIME command.
0000   ;
0000   ;       V03-019 RLRSCORPIO       Robert L. Rappaport     16-Mar-1984
0000   ;               Begin additions (to INI$IOMAP) for Scorpio support.
0000   ;               Also move ADAPDESC to SYSMAR.MAR, changing it to remove
0000   ;               the ADAP_GENERAL array.
0000   ;
0000   ;       V03-018 RLRINIADP        Robert Rappaport        28-Feb-1984
0000   ;               Add refinements to previous update that introduces
0000   ;               longword array CONFREG.  Mainly add logic to allow for
0000   ;               independently assembled invocations of ADAPDESC macro
0000   ;               to be linked into this code.  This provides possible
0000   ;               support of BI as a public bus, with user defined nodes.
0000   ;
0000   ;       V03-017 KPL0100          Peter Lieberwirth       30-Jan-1984
0000   ;               Implement first step towards a longword-array CONFREG to
0000   ;               replace current byte array CONFREG.  INIADP will construct
0000   ;               two confregs, CONFREG and CONFREGL.  CONFREGL will be
0000   ;               a longword array.  The high byte will be a VMS-bus
0000   ;               designation, and the low word will contain the 16-bit
0000   ;               device type.  The BI introduces 16 bit device types.
0000   ;
0000   ;               When all references to CONFREG have been modified to touch
0000   ;               CONFREGL, INIADP will be modified again to stop creating
0000   ;               the byte array.
0000   ;
0000   ;               While here, map 9 pages of CI register space, up from 8.
0000   ;
0000   ;       V03-016 KPL0001          Peter Lieberwirth       17-Jan-1984
0000   ;               Fix bug in V03-015 that caused a failure to boot on 750s.
0000   ;               Specifically, add NDT$_MEM1664NI to ADAPDESC macro.
0000   ;
0000   ;       V03-015 TCM0009          Trudy C. Matthews       12-Dec-1983
0000   ;               Add support for booting from VENUS console device to
0000   ;               INI$CONSOLE.  When mapping I/O space on VENUS, use the
0000   ;               PAMM to determine if any adaptors are present on the
0000   ;               ABUS.
0000   ;
```

```
0000   ;       V03-014 KDM0081         Kathleen D. Morse       13-Sep-1983
0000   ;               Create version for Micro-VAX I.
0000   ;
0000   ;       V03-013 DWT0126         David W. Thiel          30-Aug-1983
0000   ;               Modify EXE$INIT_TODR to set internal time without
0000   ;               modifying the contents of the system disk.
0000   ;
0000   ;       V03-012 KDM0062         Kathleen D. Morse       18-Jul-1983
0000   ;               Add loadable, cpu-dependent routine for initializing
0000   ;               the time-wait loop data cells, EXE$INI_TIMWAIT.
0000   ;
0000   ;       V03-011 KDM0057         Kathleen D. Morse       15-Jul-1983
0000   ;               Added loadable, cpu-dependent routine for initializing
0000   ;               the system time, EXE$INIT_TODR.
0000   ;
0000   ;       V03-010 KTA3071         Kerbey T. Altmann       12-Jul-1983
0000   ;               Include CPU-specific console init code.
0000   ;
0000   ;       V03-009 TCM0008         Trudy C. Matthews       10-Jan-1983
0000   ;               Change PSECT of 11/790 data that must stick around after
0000   ;               INIADP is deleted.  Build arrays ABUS_VA, ABUS_TYPE, and
0000   ;               ABUS_INDEX that describe the 11/790 ABUS configuration.
0000   ;
0000   ;       V03-008 MSH0002         Maryann Hinden          08-Dec-1982
0000   ;               Add powerfail support for DW750.
0000   ;
0000   ;       V03-007 ROW0142         Ralph O. Weber          24-NOV-1982
0000   ;               Change UBA interrupt services routines prototype so that
0000   ;               UBAERRADR is correctly computed as an offset from UBAINTBASE.
0000   ;
0000   ;       V03-006 TCM0007         Trudy C. Matthews       10-Nov-1982
0000   ;               Add 11/790-specific initialization of SCB.
0000   ;
0000   ;       V03-005 TCM0006         Trudy C. Matthews       8-Nov-1982
0000   ;               Initialize field ADP$L_AVECTOR with the address of
0000   ;               each adapter's first SCB vector.
0000   ;
0000   ;       V03-004 KTA3018         Kerbey T. Altmann       30-Oct-1982
0000   ;               Move from INILOA facility, rename from INITADP,
0000   ;               put in conditional assembly, rewrite some routines.
0000   ;
0000   ;       V03-003 MSH0001         Maryann Hinden          24-Sep-1982
0000   ;               Change EXE$DW780_INT to EXE$UBAERR_INT.
0000   ;
0000   ;       V03-002 TCM0005         Trudy C. Matthews       10-Aug-1982
0000   ;               Added support for 11/790 processor.
0000   ;
0000   ;       V03-001 KDM0002         Kathleen D. Morse       28-Jun-1982
0000   ;               Added $DCDEF.
0000   ;
0000   ;--
```

```
0000     ;
0000     ; MACRO LIBRARY CALLS
0000     ;
0000             $ADPDEF                         ; Define ADP offsets.
0000             $BIICDEF                        ; Define BIIC offsets.
0000             $BQODEF                         ; Define boot vector offsets.
0000             $BTDDEF                         ; Define boot devices
0000             $BUADEF                         ; Define BUA Register offsets.
0000             $CRBDEF                         ; Define CRB offsets.
0000             $DCDEF                          ; Define adapter types
0000             $DDBDEF                         ; Define DDB offsets
0000             $DYNDEF                         ; Define data structure type codes.
0000             $IDBDEF                         ; Define interrupt dispatcher offsets.
0000             $IOUV1DEF                       ; Define Micro-VAX I I/O space.
0000             $MCHKDEF                        ; Define machine check masks.
0000             $NDTDEF                         ; Define nexus device types.
0000             $PRDEF                          ; Define IPR numbers.
0000
0000
0000
0000
0000
0000             $PRUV1DEF                       ; Define Micro-VAX I specific IPRs.
0000
0000
0000             $PTEDEF                         ; Define Page Table Entry bits.
0000             $RPBDEF                         ; Define Restart Parameter Block fields.
0000             $UBADEF                         ; Define UBA register offsets.
0000             $UCBDEF                         ; Define UCB offsets.
0000             $VADEF                          ; Define virtual address fields.
0000             $VECDEF                         ; Define vec offsets.
```

```
0000          .SBTTL  Macros to describe nexus configurations
0000    ;
0000    ;     The macros FLOAT_NEXUS and FIXED_NEXUS add one or more entries to a
0000    ;     nexus descriptor table.  Each entry is of the form:
0000    ;                +--------------------------------+
0000    ;                |   PFN of nexus I/O space       |
0000    ;                +--------+--------+--------------+
0000    ;                |  bus   |   0    |     type     |
0000    ;                +--------+--------+--------------+
0000    ;     type = 0 -> floating nexus
0000    ;     type = non-zero -> fixed nexus; type = fixed adapter type
0000    ;     bus  = 0, if SBI; %x80 if BI  (this is a VMS-only designation)
0000    ;
0000    ;
0000    ;     device_type:      SBI adapters have 8-bit device type codes.  These
0000    ;                       device types are simple integers.
0000    ;
0000    ;                       BI adapters have 16-bit device type codes, that are
0000    ;                       subject to the following interpretation:
0000    ;
0000    ;                       - the MSB of the device-type field will be 0 for DEC
0000    ;                       devices and 1 for non-DEC devices.
0000    ;
0000    ;                       - DEC memory devices will have 0s in the high-order
0000    ;                       byte of the device type.
0000    ;
0000    ;                       - non-DEC supplied memory devices will have a 1 in the
0000    ;                       MSB of the high-order byte, and the rest of the high
0000    ;                       order byte will contain 0s.
0000    ;
0000    ;                       - The "all 0s" and "all 1s" device-type codes are
0000    ;                       reserved for DEC.
0000    ;
0000    ; If SBI type codes were simply expanded to a word for purposes of the routines
0000    ; in this module, there would be possible conflicts between SBI devices and
0000    ; BI memory adapters supplied by DEC.  Voila:  the bus type.
0000    ;
0000    ; Macro FLOAT_NEXUS.
0000    ; INPUTS:
0000    ;       PHYSADR -- physical address of 1 or more contiguous floating nexus
0000    ;                  slots
0000    ;       NUMNEX -- number of contiguous floating nexuses, default = 1
0000    ;       PERNEX -- amount of address space per nexus (does not have to be
0000    ;                 specified if NUMNEX = 1)
0000    ;
0000          .MACRO  FLOAT_NEXUS      PHYSADR,NUMNEX=1,PERNEX=0
0000          PA = PHYSADR
0000          .REPEAT NUMNEX           ; For each nexus...
0000          .LONG   <PA/^X200>       ; Store PFN.
0000          .LONG   0                ; Store floating nexus type.
0000          PA = PA + PERNEX         ; Increment to physical address of next nexus.
0000          .ENDR
0000          .ENDM   FLOAT_NEXUS
0000
0000    ;
0000    ; Macro FIXED_NEXUS.
0000    ;
```

```
          0000  ; INPUTS:
          0000  ;       PHYSADR - physical address of 1 or more contiguous fixed nexus slots
          0000  ;       PERNEX - amount of address space per nexus
          0000  ;       NEXUSTYPES - a list of fixed nexus types, enclosed in <>
          0000  ;
          0000          .MACRO  FIXED_NEXUS     PHYSADR,PERNEX=0,NEXUSTYPES
          0000          PA = PHYSADR
          0000          .IRP    TYPECODE,NEXUSTYPES     ; For each fixed nexus type...
          0000          .LONG   <PA/^X200>              ; Store PFN.
          0000          .LONG   TYPECODE                ; Store fixed nexus type.
          0000          PA = PA + PERNEX                ; Increment to address of next nexus.
          0000          .ENDR
          0000          .ENDM   FIXED_NEXUS
          0000
          0000  ;
          0000  ; Macro NEXUSDESC_TABLE - declare the beginning of a NEXUS descriptor table
          0000  ;
          0000  ;       1st byte in table (at offset -5 from label) contains length of
          0000  ;       adapter type code field in CSR's on this bus. [Note for SBI like
          0000  ;       busses, this is 1.]  The next longword (at offset -4) in the
          0000  ;       table contains the Software defined bus type byte defined in the
          0000  ;       high order byte of the longword.  [Note for SBI like busses, this
          0000  ;       value is 0, for the BI it is ^x80.]
          0000  ;
          0000
          0000  ; Define parameters that may be specified or used in macro invocation.
          0000
00000000  0000  BI_LIKE  = 0                            ; BI like bus.
00000001  0000  SBI_LIKE = 1                            ; SBI like bus.
          0000
00000001  0000  SBI_CSR_LEN = 1                         ; Length of type code field in adapter CSR's
          0000                                          ;  on SBI, CMI, etc.
00000002  0000  BI_CSR_LEN  = 2                         ; Length of type code field in adapter CSR's
          0000                                          ;  on BI.
          0000
00000000  0000  SBI_BUS_CODE = 0                        ; Software defined bus code for SBI like busses.
80000000  0000  BI_BUS_CODE  = ^x80000000               ; Software defined bus code for the BI.
          0000
          0000          .MACRO  NEXUSDESC_TABLE LABEL,BUS_TYPE=SBI_LIKE
          0000          .IF     EQ,BUS_TYPE-SBI_LIKE
          0000                                          .BYTE   SBI_CSR_LEN
          0000                                          .LONG   SBI_BUS_CODE
          0000          .IFF
          0000                  .IF     EQ,BUS_TYPE-BI_LIKE
          0000                                          .BYTE   BI_CSR_LEN
          0000                                          .LONG   BI_BUS_CODE
          0000                  .IFF
          0000                                          .ERROR  ; UNRECOGNIZED BUS TYPE, NEXUSDESC_TABLE;
          0000                  .ENDC
          0000          .ENDC
          0000
          0000  LABEL:
          0000          .ENDM   NEXUSDESC_TABLE
          0000
FFFFFFFB  0000  CSR_LEN_OFFSET  = -5                      ; Offset before nexus descriptor of
          0000                                            ;  byte containing length of adapter
          0000                                            ;  type field in adapter CSR.
```

```
FFFFFFFC  0000   BUS_CODE_OFFSET = -4                        ; Offset before nexus descriptor table
          0000                                               ;  of longword containing software
          0000                                               ;  defined bus type to be or'ed with
          0000                                               ;  adapter type to produce NDT$_ value.
          0000   ;
          0000   ; Macro END_NEXUSDESC.
          0000   ;
          0000           .MACRO  END_NEXUSDESC
          0000           .LONG   0                           ; PFN=0 -> end of nexus descriptors.
          0000           .ENDM   END_NEXUSDESC
```

```
             0000          .SBTTL  Adapter-specific data structures
             0000  ;
             0000  ; Put a symbol for arrays built by macros in the correct psects.
             0000  ;
             0000  ;***************** ADAPTERS array *************
         00000000          .PSECT  $$$INIT$DATA0
             0000  ADAPTERS:                              ; Build adapter type code arrays here.
             0000
         00000000          .PSECT  $$$INIT$DATA1          ; User contributions in this .PSECT.
             0000                                         ; End of ADAPTERS array.
             0000  ;***************** End of ADAPTERS array *************
             0000
             0000  ;***************** NUM_PAGES array *************
         00000000          .PSECT  $$$INIT$DATA2
             0000  NUM_PAGES:                             ; Build "number of pages to map" array.
         00000000          .PSECT  $$$INIT$DATA3          ; User contributions in this .PSECT.
             0000  ;***************** End of NUM_PAGESarray *************
             0000
             0000  ;***************** INIT_ROUTINES array *************
         00000000          .PSECT  $$$INIT$DATA4
             0000  INIT_ROUTINES:                         ; Build "address of init routine" array.
         00000000          .PSECT  $$$INIT$DATA5          ; User contributions in this .PSECT.
             0000  ;***************** End of INIT_ROUTINES array *************
             0000
             0000  ;
             0000  ; To add a new adapter type:
             0000  ;       1) Add a new ADAPDESC macro invocation to the end of this list.
             0000  ;
         00000000          .PSECT  $$$INIT$DATA,LONG
             0000
             0000  ;
             0000  ; Default interupt vectors for UNIBUS system devices
             0000  ; (This array is indexed by the RPB field RPB$B_DEVTYP, if the RPB field
             0000  ; RPB$W_ROUBVEC is zero.  If RPB$W_ROUBVEC is not zero, then RPB$W_ROUBVEC
             0000  ; is used and this array is not referenced at all.  RPB$W_ROUBVEC is set up
             0000  ; by PQDRIVER.  RPB$L_BOOTR0 is set by VMB to contain the device name in
             0000  ; ASCII, not the vector number and device type, as it does on full
             0000  ; architecture VAX machines.
             0000  ;
             0000  BOOTVECTOR:
     0088    0000          .WORD   ^X88                   ; RK06/7 Interrupt vector
     0070    0002          .WORD   ^X70                   ; RL01/2 Interrupt vector
             0004
             0004  BUS_CSR_LEN:                           ; Static byte containing the length (in bytes)
       00    0004          .BYTE   0                      ;  of the adapter type field in the CSR's of
             0005                                         ;  the bus currently being configured.  The
             0005                                         ;  proper value for the bus of interest is
             0005                                         ;  copied here, from the current nexus
             0005                                         ;  descriptor table, when we enter subroutine
             0005                                         ;  CONFIG_IOSPACE.
             0005
             0005  SW_BUS_CODE:                           ; Static longword containing the software
 00000000    0005          .LONG   0                      ;  defined bus type, of the bus currently being
             0009                                         ;  configured, in the high order byte.  The
             0009                                         ;  proper value for the bus of current interest
             0009                                         ;  is copied here, from the nexus descriptor
             0009                                         ;  table, when we enter subroutine
```

```
                 0009                                            ;  CONFIG_IOSPACE.
                 0009
                 0009  DIRECT_VEC_NODE_CNT:                      ; Static longword that counts the number of
                 0009                                            ;  direct vectoring adpater nodes that we have
       00000000  0009          .LONG   0                         ;  run across so far.
                 000D
       00000001  000D  $$$VMSDEFINED = 1                         ; Define symbol that means VMS system software.
       00000080  000D  NUMUBAVEC = 128                           ; ALLOW FOR 128 UNIBUS VECTORS
                 000D
                 000D                  ADAPDESC -                ; Memory. ** MUST BE 1ST IN DESCRIPTOR LIST **
                 000D                  ADPTYPES=<NDT$_MEM1664NI,NDT$_MEM4NI,NDT$_MEM4I,NDT$_MEM16NI, -
                 000D                          NDT$_MEM16I, -
                 000D                          NDT$_MEM64NIL,NDT$_MEM64EIL,NDT$_MEM64NIU,NDT$_MEM64EIU,  -
                 000D                          NDT$_MEM64I, -
                 000D                          NDT$_MEM256NIL,NDT$_MEM256EIL,NDT$_MEM256NIU,NDT$_MEM256EIU,  -
                 000D                          NDT$_MEM256I, -
                 000D                          NDT$_SCORMEM> -
                 000D                          NUMPAGES=1
                 000D
                 000D                  ADAPDESC -                ; MASSbus.
                 000D                          ADPTYPES=NDT$_MB, -
                 000D                          NUMPAGES=8, -
                 000D                          INITRTN=INI$MBADP
                 000D
                 000D                  ADAPDESC -                ; UNIbus.
                 000D                          ADPTYPES=<NDT$_UB0,NDT$_UB1,NDT$_UB2,NDT$_UB3,NDT$_BUA>, -
                 000D                          NUMPAGES=8, -
                 000D                          INITRTN=INI$UBSPACE
                 000D
                 000D                  ADAPDESC -                ; Multi-port memory.
                 000D                          ADPTYPES=<NDT$_MPM0,NDT$_MPM1,NDT$_MPM2,NDT$_MPM3>, -
                 000D                          NUMPAGES=1, -
                 000D                          INITRTN=INI$MPMADP
                 000D
                 000D                  ADAPDESC -                ; DR32.
                 000D                          ADPTYPES=NDT$_DR32, -
                 000D                          NUMPAGES=4, -
                 000D                          INITRTN=INI$DRADP
                 000D
                 000D                  ADAPDESC -                ; CI780
                 000D                          ADPTYPES=NDT$_CI, -
                 000D                          NUMPAGES=9, -
                 000D                          INITRTN=INI$CIADP
                 000D
                 000D                  ADAPDESC -                ; KDZ11 Processor
                 000D                          ADPTYPES=NDT$_KDZ11, -
                 000D                          NUMPAGES=1, -
                 000D                          INITRTN=INI$KDZ11
                 000D
```

```
                    000D     523          .SBTTL  CPU-specific data structures
                    000D     524  ;
                    000D     525  ; To add a new CPU type:
                    000D     526  ;       1) Create a new nexus descriptor table, using FLOAT_NEXUS and
                    000D     527  ;          FIXED_NEXUS macros.  Put an END_NEXUSDESC macro at the end.
                    000D     528  ;
                    000D     529
                    000D     552
                    000D     590
                    000D     617
                    000D     659
                    000D     660
                    000D     662 CPU_ADPSIZE:
              0258  000D     663          .WORD   ADP$C_UBAADPLEN
                    000F     664
                    000F     665
                    000F     666  ;
                    000F     667  ; Declare the beginning of a nexus-descriptor table.
                    000F     668  ;
                    000F     669          NEXUSDESC_TABLE LABEL=NEXUSDESC
                    0014     670
                    0014     671
                    0014     672  ;
                    0014     673  ; Describe all nexuses on a Micro-VAX I processor.
                    0014     674  ;
          00000000  0014     675          SBI_CPU = 0
          00000000  0014     676          BI_CPU  = 0
                    0014     677          FIXED_NEXUS -
                    0014     678                  PHYSADR=IOUV1$AL_QBOSP, -
                    0014     679                  NEXUSTYPES=NDT$_UB0
                    001C     680          END_NEXUSDESC
                    0020     682
                    0020     706
                    0020     707  ;
                    0020     708  ; Nexus "descriptor" arrays -- these arrays hold the nexus-device type and
                    0020     709  ; virtual address of every adapter on the system.  The arrays, CONFREGL and
                    0020     710  ; SBICONF, are allocated enough space to hold the maximum number of adapters
                    0020     711  ; that can be attached to any CPU.  When the code discovers how many adapters
                    0020     712  ; actually exist on the system, it will allocate space from non-paged pool
                    0020     713  ; and move a permanent copy of these arrays into that space.
                    0020     714  ;
          00000040  0020     715 MAXNEXUS = 64
                    0020     716 CONFREG:                              ; Byte array of nexus-device type codes..
          00000060  0020     717          .BLKB   MAXNEXUS
                    0060     718 SBICONF:
          00000160  0060     719          .BLKL   MAXNEXUS              ; Longword array of VAs of adapter space.
                    0160     720 CONFREGL:
          00000260  0160     721          .BLKL   MAXNEXUS              ; Longword array of nexus-device type codes
```

```
                                      0260     723           .SBTTL  Message strings
                                      0260     724
                            0000000D  0260     725 CR = 13
                            0000000A  0260     726 LF = 10
                                      0260     727 NOSPT:
2D 54 49 4E 49 43 45 58 45 25 0A 0D   0260     728           .ASCIZ  <CR><LF>/%EXECINIT-F-Insufficient SPT entries/<CR><LF>
65 69 63 69 66 66 75 73 6E 49 2D 46   026C
69 72 74 6E 65 20 54 50 53 20 74 6E   0278
                  00 0A 0D 73 65      0284
```

```
                          0289          .SBTTL  INI$IOMAP, Initialize and map nexuses
                          0289  ;++
                          0289  ; FUNCTIONAL DESCRIPTION:
                          0289  ;       This routine is executed only once, during system initialization.
                          0289  ;       It loops through all nexuses on the system, testing for
                          0289  ;       adapters.  When it finds an adapter, it maps its I/O space and
                          0289  ;       initializes it.
                          0289  ;
                          0289  ; INPUTS:
                          0289  ;       BOO$GL_SPTFREL  - next free VPN
                          0289  ;       MMG$GL_SPTVASE  - base of system page table
                          0289  ;       EXE$GL_RPB      - address of reboot parameter block
                          0289  ;
                          0289  ; OUTPUTS:
                          0289  ;       R0 - SS$_NORMAL
                          0289  ;
                          0289  ;       For each adapter found, its accessible I/O space is mapped to virtual
                          0289  ;       addresses.  An ADP (Adapter Control Block) is built, and the hardware
                          0289  ;       adapter is initialized.
                          0289  ;
                          0289  ;       The arrays CONFREG (a byte array of nexus-device type codes, defined
                          0289  ;       by NDT$_ symbols) and SBICONF (a longword array of
                          0289  ;       virtual addresses that map adapter space) are initialized.  Pointers
                          0289  ;       to these arrays are stored in EXE$GL_CONFREG  and
                          0289  ;       MMG$GL_SBICONF.  The number of entries in these two parallel arrays is
                          0289  ;       stored in EXE$GL_NUMNEXUS.
                          0289  ;
                          0289  ;       Since BI devices have a 16-bit device type code, a new CONFREG array is
                          0289  ;       constructed.  This is a longword array called CONFREGL.
                          0289  ;
                          0289  ;       Several locations in the RPB that describe the boot device are init'ed:
                          0289  ;       RPB$L_BOOTR1    - 'olds index into CONFREG and SBICONF for the boot
                          0289  ;                         adapter
                          0289  ;       RPB$L_ADPVIR    - holds VA of boot device adapter's register space
                          0289  ;       RPB$L_CSRVIR    - holds VA of boot device's register space
                          0289  ;--
                          0289
                      00000000          .PSECT  $$$INIT$CODE,QUAD
                          0000  INI$IOMAP::
                          0000
          0FFF 8F     BB  0000          PUSHR   #^M<R0,R1,R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
                          0004  ;
                          0004  ; Set up common inputs to CONFIG_IOSPACE subroutine for the CPU-specific code.
                          0004  ;
     52   00000000'GF DO  0004          MOVL    G^BOO$GL_SPTFREL,R2      ; Get next available VPN.
     53   00000000'GF DO  000B          MOVL    G^MMG$GL_SPTBASE,R3      ; Get base of System Page Table.
            53   6342 DE  0012          MOVAL   (R3)[R2],R3              ; Compute SVASPT.
         52   52   09 78  0016          ASHL    #9,R2,R2                 ; Convert VPN to VA.
     52   80000000 8F C8  001A          BISL    #VA$M_SYSTEM,R2          ; Set system bit.
                   54 D4  0021          CLRL    R4                       ; Clear index into CONFREG and SBICONF.
     59   00000000'GF DO  0023          MOVL    G^EXE$GL_RPB,R9          ; Get address of RPB.
00000000'GF   0060'CF DE  002A          MOVAL   W^SBICONF,G^MMG$GL_SBICONF   ; Set pointers to local copies
00000000'GF   0020'CF DE  0033          MOVAL   W^CONFREG,G^EXE$GL_CONFREG   ; of these arrays for init routines.
00000000'GF   0160'CF DE  003C          MOVAL   W^CONFREGL,G^EXE$G[_CONFREGL ; ...
```

```
                          0045          .SBTTL  INITADP_780, _750, _730, and _UV1
                          0045     ;
                          0045     ; I/O address space for the 11/780, 11/750, 11/730, and Micro-VAX I cpus
                          0045     ; is statically defined in their respective nexus descriptor tables.
                          0045     ;
   56    0014'CF      DE  0045          MOVAL   W^NEXUSDESC,R6          ; Get address of nexus table.
               5B     D4  004A          CLRL    R11                     ; Signal use 1st page of SCB.
               0B     10  004C          BSBB    CONFIG_IOSPACE          ; Configure processor I/O space.
                          004E
                          004E
             0079     30  004E          BSBW    CREATE_ARRAYS           ; Create CONFREG and SBICONF arrays.
          0FFF 8F     BA  0051          POPR    #^M<R0,R1,R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
          50   01     D0  0055          MOVL    #1,R0                   ; Set success status
                      05  0058          RSB                             ; Return.
```

```
                              0059   916          .SBTTL  CONFIG_IOSPACE
                              0059   917  ;
                              0059   918  ; CONFIG_IOSPACE
                              0059   919  ;        Given a nexus descriptor table, which describes what "nexuses" or
                              0059   920  ;        "slots" are available on a system to hold I/O adapters, find and
                              0059   921  ;        initialize all adapters on the system.
                              0059   922  ;
                              0059   923  ; Inputs:
                              0059   924  ;        R2 - next available virtual address, to be used for mapping I/O space
                              0059   925  ;        R3 - address of PTE associated with VA in R2
                              0059   926  ;        R4 - Current index into CONFREG and SBICONF arrays (should be 0 the
                              0059   927  ;             first time CONFIG_IOSPACE is called)
                              0059   928  ;        R6 - address of nexus descriptor table
                              0059   929  ;        R9 - address of Restart Parameter Block (RPB)
                              0059   930  ;        R10 - PFN of boot adapter space
                              0059   931  ;        R11- page offset from beginning of SCB; tells which page of the SCB
                              0059   932  ;             to use for this set of nexuses (passed to routines that init ADP)
                              0059   933  ;
                              0059   934  ; Outputs:
                              0059   935  ;        R2,R3,R4 - updated
                              0059   936  ;        R9,R10,R11 - preserved; all other registers potentially modified
                              0059   937  ;        CONFREG - initialized with adapter NDT$ code for each nexus
                              0059   938  ;        SBICONF - initialized with adapter space VA for each nexus
                              0059   939  ;
                              0059   940 CONFIG_IOSPACE:
                              0059   946  ;
                              0059   947  ; There is only one adapter, the Qbus.
                              0059   948  ;
                              0059   950
           FB A6    90        0059   951          MOVB    CSR_LEN_OFFSET(R6),-    ; Move length of adapter type field
         0004'CF              005C   952                  W^BUS_CSR_LEN           ;  in CSR's to static location.
           FC A6    D0        005F   953          MOVL    BUS_CODE_OFFSET(R6),-   ; Move software defined bus type code
         0005'CF              0062   954                  W^SW_BUS_CODE           ;  to static longword.
                              0065   955
                              0065   956 NXT_NEXUS:                              ; For each nexus...
           58  86   D0        0065   957          MOVL    (R6)+,R8                ; Get PFN of nexus.
                              0068   984  ;
                              0068   985  ; Execution continues here if adapter was present.
                              0068   986  ;
                              0068   987 GET_TYPE:
           57  86   D0        0068   988          MOVL    (R6)+,R7                ; Get nexus-device type from nexus table.
                              006B  1005  ;
                              006B  1006  ; Here R7 has hardware adapter code or'ed with software bus code.
                              006B  1007  ; Translate specific nexus device type code into general adapter type code.
                              006B  1008  ;
                              006B  1009 GET_GEN_TYPE:
  0020'CF44    57   90        006B  1010          MOVB    R7,W^CONFREG[R4]        ; Save nexus-device type in CONFREG.
  0160'CF44    57   D0        0071  1011          MOVL    R7,W^CONFREGL[R4]       ; CONFREGL also filled in.
               55   D4        0077  1012          CLRL    R5                      ; Clear loop index.
                              0079  1013 30$:                                    ;
  50   0000'CF45    DE        0079  1014          MOVAL   W^ADAPTERS[R5],R0       ; Get address of adapter type code.
         0000'CF    9F        007F  1015          PUSHAB  W^NUM_PAGES             ; Push addr of end of ADAPTERS array.
           8E  50   D1        0083  1016          CMPL    R0,(SP)+                ; See if we went beyond array.
               3F   1E        0086  1017          BGEQU   END_NEXUS               ; unrecognized adapter, do not map.
           60  57   D1        0088  1018          CMPL    R7,(R0)                 ; Adapter type match?
               04   13        008B  1019          BEQL    40$                     ; If EQL yes, adapter type match.
               55   D6        008D  1020          INCL    R5                      ; Increment loop index.
```

```
                E8   11  008F           BRB     30$                         ; Look at next adapter.
                         0091   40$:
                         0091
                         0091   ;
                         0091   ; Store boot parameters.
                         0091   ;
           60 A9   52  D0  0091           MOVL    R2,RPB$L_ADPVIR(R9)         ; Store VA of boot adapter space.
           20 A9   54  D0  0095           MOVL    R4,RPB$L_BOOTR1(R9)         ; Store boot adapter nexus number.
   51   54 A9   0D   00  EF  0099           EXTZV   #0,#13, -                   ; Get offset into UNIBUS/QBUS I/O page.
                         009F                   RPB$L_CSRPHY(R9),R1         ;
    58 A9    1000 C241   9E  009F           MOVAB   <8*512>(R2)[R1], -          ; Set VA of UNIBUS/QBUS registers.
                         00A6                   RPB$L_CSRVIR(R9)            ;
                         00A6
                         00A6
                         00A6   ;
                         00A6   ; R5/ general adapter type; index into "general" adapter arrays.
                         00A6   ; For each adapter -
                         00A6   ;       Map the # of pages specified in ADAPDESC macro
                         00A6   ;       JSB to initialization routine specified in ADAPDESC macro
                         00A6   ;
                         00A6   MAP_NEXUS:
       58    2000 C8   9E  00A6           MOVAB   <16*512>(R8),R8             ; Since no Qbus adapter space, point to
                         00AB                                               ;  non-exist memory past Qbus I/O space.
    0060'CF44    52   D0  00AB           MOVL    R2,W^SBICONF[R4]            ; Save VA of adapter space in SBICONF.
    51    0000'CF45   3C  00B1           MOVZWL  W^NUM_PAGES[R5],R1          ; Get number of pages to map.
                  6A   10  00B7           BSBB    MAP_PAGES                   ; Map the I/O pages.
    51    0000'CF45   DE  00B9           MOVAL   W^INIT_ROUTINES[R5],R1      ; Get address of initialization routine.
                  61   D5  00BF           TSTL    (R1)                        ; Initialization routine specified?
                  04   13  00C1           BEQL    END_NEXUS                   ; Branch if none.
             00 B141   16  00C3           JSB     @(R1)[R1]                   ; Call initialization routine.
                         00C7   END_NEXUS:
                  54   D6  00C7           INCL    R4                          ; Increment CONFREG and SBICONF index.
                       05  00C9           RSB                                 ; Return, as only one nexus.
                         00CA
```

```
                             00CA                   .SBTTL  CREATE_ARRAYS
                             00CA      ;
                             00CA      ; CREATE_ARRAYS
                             00CA      ;
                             00CA      ;            Move the local CONFREG and SBICONF arrays into non-paged pool.
                             00CA      ;
                             00CA      ; Inputs:
                             00CA      ;            R4 - Number of nexuses on the system.
                             00CA      ;            CONFREG and SBICONF have been initialized.
                             00CA      ;
                             00CA      ; Outputs:
                             00CA      ;            R0 - R5 destroyed
                             00CA      ;            EXE$GL_CONFREG points to a copy of the CONFREG array in non-paged pool
                             00CA      ;            MMG$GL_SBICONF points to a copy of the SBICONF array in non-paged pool
                             00CA      ;            EXE$GL_NUMNEXUS contains the number of nexuses on the system
                             00CA      ;
                             00CA      ;
                             00CA      CREATE_ARRAYS:
    00000000'GF     54   D0  00CA               MOVL    R4,G^EXE$GL_NUMNEXUS      ; Store number of nexuses on system.
         51   0C  A444   DE  00D1               MOVAL   12(R4)[R4],R1             ; Allocate n bytes for CONFREG plus
                             00D6                                                 ; 4n bytes for SBICONF + header
            51    6144   DE  00D6               MOVAL   (R1)[R4],R1               ; Another 4n bytes for CONFREGL.
                  017D   30  00DA               BSBW    ALONPAGD                  ; Get pool for CONFREG and SBICONF.
                    82   7C  00DD               CLRQ    (R2)+                     ; Clear out unused
               82   51   B0  00DF               MOVW    R1,(R2)+                  ; Set in size
          82   0763 8F   B0  00E2               MOVW    #<DYN$C_CONF@8>!DYN$C_INIT,(R2)+ ; Set type and subtype
    00000000'GF     62   9E  00E7               MOVAB   (R2),G^EXE$GL_CONFREG     ; Store address of system CONFREG.
            51    6244   9E  00EE               MOVAB   (R2)[R4],R1               ; Two steps to CONFREGL, 1st, SBICONF,
    00000000'GF     51   D0  00F2               MOVL    R1,G^MMG$GL_SBICONF       ; Store address of system SBICONF.
   00000000'GF    6144   DE  00F9               MOVAL   (R1)[R4],G^EXE$GL_CONFREGL ; And address of system CONFREGL.
                    14   BB  0101               PUSHR   #^M<R2,R4>                ; Save pool address and nexus count.
     62    0020'CF  54   28  0103               MOVC3   R4,W^CONFREG,(R2)         ; Copy CONFREG to pool.
                    14   BA  0109               POPR    #^M<R2,R4>                ; Retrieve pool address and nexus count.
         51   54    04   C5  010B               MULL3   #4,R4,R1                  ; Number of bytes in SBICONF.
              7E    51   D0  010F               MOVL    R1,-(SP)                  ; Save, SBICONF size = CONFREGL size
   6244    0060'CF  51   28  0112               MOVC3   R1,W^SBICONF,(R2)[R4]     ; Copy SBICONF to pool.
              51    8E   D0  0119               MOVL    (SP)+,R1                  ; Restore size of SBICONF and CONFREGL.
     63    0160'CF  51   28  011C               MOVC3   R1,W^CONFREGL,(R3)        ; Copy CONFREGL to pool.  R3 is output
                             0122                                                 ; from SBICONF MOVC3, so SBICONF and
                             0122                                                 ; CONFREGL must be adjacent.
                             0122
                         05  0122               RSB
```

```
                                    0123         .SBTTL  MAP_PAGES
                                    0123  ;++
                                    0123  ; INPUTS:
                                    0123  ;         R1/ Number of pages to map.
                                    0123  ;         R2/ VA of page to map.
                                    0123  ;         R3/ VA of system page table entry to be used.
                                    0123  ;         R8/ PFN of page(s) to map.
                                    0123  ;
                                    0123  ; OUTPUTS:
                                    0123  ;         R2,R3 updated; R1,R8 destroyed; all other registers preserved
                                    0123  ;
                                    0123  ;--
                                    0123
                                    0123  MAP_PAGES:
                                    0123
      83    58   90000000 8F    C9  0123         BISL3   #<PTE$M_VALID!PTE$C_KW>,R8,(R3)+
                                    012B                                                 ; Map a page.
                             58 D6  012B         INCL    R8                              ; Next PFN.
                   52   0200 C2 9E  012D         MOVAB   512(R2),R2                      ; Next VA.
                     00000000'GF D6  0132         INCL    G^BOO$GL_SPTFREL                ; Check for no more system page
   00000000'GF       00000000'GF D1  0138         CMPL    G^BOO$GL_SPTFREH, -             ; Check for no more system page
                                    0143                 G^BOO$GL_SPTFREL                ; table entries.
                             04 15  0143         BLEQ    ERROR_HALT                      ; Branch if out of SPTEs.
                          DB 51 F5  0145         SOBGTR  R1,MAP_PAGES                    ; Map another page.
                                05  0148         RSB                                     ; All done.
                                    0149
                                    0149  ERROR_HALT:
                   51    0260'CF 9E  0149         MOVAB   W^NOSPT,R1                      ; Set error message.
                                    014E  ERROR_HALT_1:
                             5B D4  014E         CLRL    R11                             ; Indicate console terminal.
                     00000000'GF 16  0150         JSB     G^EXE$OUTZSTRING                ; Output error message.
                                00  0156         HALT                                    ; ***** FATAL ERROR *******
```

```
                                    0157  1269          .SBTTL  INI$UBSPACE
                                    0157  1270  ;++
                                    0157  1271  ;       Map UNIBUS space; initialize UNIBUS ADP.
                                    0157  1272  ;
                                    0157  1273  ; INPUTS:
                                    0157  1274  ;       R2 - VA of next free system page
                                    0157  1275  ;       R3 - VA of system page table entry to be used to map VA in R2
                                    0157  1276  ;       R4 - nexus identification number of this adapter
                                    0157  1277  ;   -8(R6) - PFN of this UNIBUS adapter's register space
                                    0157  1278  ;
                                    0157  1279  ; OUTPUTS:
                                    0157  1280  ;       UNIBUS space is mapped.
                                    0157  1281  ;       INI$UBADP is called to build an ADP block and initialize UNIBUS
                                    0157  1282  ;       adapter hardware.
                                    0157  1283  ;
                                    0157  1284  ;--
                                    0157  1285
                                    0157  1286  INI$UBSPACE:
                                    0157  1287
         58    0160'CF44      DE    0157  1290          MOVAL   W^CONFREGL[R4],R8               ; R8 => CONFREGL slot.
      58  68     02    00     EF    015D  1291          EXTZV   #0,#2,(R8),R8                   ; Get UBA number.
          58     58    09     78    0162  1292          ASHL    #9,R8,R8                        ; Position UB number.
                                    0166  1295
                                    0166  1304
                                    0166  1309
                                    0166  1314
                                    0166  1319
                                    0166  1325
  58    00100000 8F    58     C3    0166  1327          SUBL3   R8,#<IOUV1$AL_QBOSP/^X200>,R8
                                    016E  1328                                          ; Get PFN of Qbus I/O page.
                                    016E  1330
                 51    10     D0    016E  1331          MOVL    #16,R1                  ; Number of pages to map (UB/Qbus space).
                     FFAF     30    0171  1332          BSBW    MAP_PAGES               ; Map I/O pages.
                                    0174  1333  ;
                                    0174  1334  ; Call adapter initialization routine.
                                    0174  1335  ;
                                    0174  1336  ;       BSBW    INI$UBADP               ; Init ADP block.
                                    0174  1337  ;       RSB
```

```
                              0174   1339         .SBTTL  INI$UBADP - BUILD ADP AND INITIALIZE UBA
                              0174   1340 ;+
                              0174   1341 ; INI$UBADP ALLOCATES AND FILLS IN AN ADAPTER CONTROL BLOCK, INTERRUPT
                              0174   1342 ; DISPATCHER AND CONNECTS THEM TO THE PROPER SCB VECTORS.  A CALL IS
                              0174   1343 ; THEN MADE TO UBA$INITIAL TO INITIALIZE THE ADAPTER HARDWARE.
                              0174   1344 ;
                              0174   1345 ; INPUT:
                              0174   1346 ;        R4 - nexus identification number of this adapter
                              0174   1347 ;        R11- offset from beginning of SCB to correct SCB page for this adapter
                              0174   1348 ;-
                              0174   1349
                              0174   1350 INI$UBADP:
                              0174   1351
        01FF 8F          BB   0174   1352         PUSHR   #^M<R0,R1,R2,R3,R4,R5,R6,R7,R8> ; SAVE R0-R8
                              0178   1353 ;
                              0178   1354 ; Allocate and initialize Adapter Control Block (ADP).
                              0178   1355 ;
   51   000D'CF          3C   0178   1356         MOVZWL  W^CPU_ADPSIZE,R1              ; PICK UP LENGTH OF ADP
           00DA          30   017D   1357         BSBW    ALONPAGD                      ; ALLOCATE SPACE FOR ADP
      08 A2   51         B0   0180   1358         MOVW    R1,ADP$W_SIZE(R2)             ; SET SIZE INTO ADP BLOCK
      0A A2   01         90   0184   1359         MOVB    #DYN$C_ADP, -                 ; AND SET TYPE OF BLOCK
                              0188   1360                 ADP$B_TYPE(R2)
      0E A2   01         B0   0188   1361         MOVW    #AT$_UBA, -                   ; SET TYPE OF ADAPTER
                              018C   1362                 ADP$W_ADPTYPE(R2)
  62    0060'CF44        D0   018C   1363         MOVL    W^SBICONF[R4], -              ; SET VA OF CONFIGURATION REG
                              0192   1364                 ADP$L_CSR(R2)
      0C A2   54         B0   0192   1365         MOVW    R4,ADP$W_TR(R2)               ; SET TR NUMBER FOR ADAPTER
                              0196   1366
      50    14 A2        DE   0196   1367         MOVAL   ADP$L_DPQFL(R2),R0            ; ADDRESS OF DATA PATH WAIT QUEUE
         60    50        D0   019A   1368         MOVL    R0,(R0)                       ; INIT QUEUE HEADER
      04 A0    50        D0   019D   1369         MOVL    R0,4(R0)                      ;
                              01A1   1370
      50    30 A2        DE   01A1   1371         MOVAL   ADP$L_MRQFL(R2),R0            ; ADDRESS OF MAP WAIT QUEUE
         60    50        D0   01A5   1372         MOVL    R0,(R0)                       ; INIT QUEUE HEADER
      04 A0    50        D0   01A8   1373         MOVL    R0,4(R0)                      ;
            04 A2        D4   01AC   1374         CLRL    ADP$L_LINK(R2)                ; ZAP ADAPTER CHAIN LINK
            FE4E'        30   01AF   1375         BSBW    ADPLINK                       ; LINK ADP TO END OF LIST
                              01B2   1376 ;
                              01B2   1377 ; Initialize adapter interrupt vectors in System Control Block.
                              01B2   1378 ;
58  00000000'GF          D0   01B2   1379         MOVL    G^EXE$GL_SCB,R8               ; GET SCB ADDRESS
                              01B9   1380
                              01B9   1387
                              01B9   1447
                              01B9   1507
                              01B9   1508
                              01B9   1536
                              01B9   1537
                              01B9   1539                                               ; REMAINING ADP INIT FOR MICRO-VAX I:
        0200 C8          DE   01B9   1540         MOVAL   ^X200(R8),-                   ; ASSUME UB0
           10 A2              01BD   1541                 ADP$L_VECTOR(R2)              ;  VECTOR SPACE
                              01BF   1542
      60 A2   0E         B0   01BF   1543         MOVW    #^XE,ADP$W_DPBITMAP(R2) ; MARK DATAPATHS 1-3 AVAILABLE
53  00000001'GF          DE   01C3   1544         MOVAL   G^UBA$UNEXINT+1,R3            ; GET ADDR OF UNEXP INT SERVICE
                              01CA   1545                                               ;  (+1 MEANS HANDLE ON INT STACK)
   54   0001'CF          DE   01CA   1546         MOVAL   W^UBA$INT0+1,R4               ; SPECIAL CASE TO COUNT PASSIVE RELEASE
                              01CF   1547
```

```
                          01CF   1548 ;
                          01CF   1549 ; INIT QBUS VECTORS TO UNEXPECTED INTERRUPT SERVICE
                          01CF   1550 ;
        50   10 A2    D0  01CF   1551          MOVL    ADP$L_VECTOR(R2),R0       ; GET ADDRESS OF VECTORS
           80   54    D0  01D3   1552          MOVL    R4,(R0)+                  ; SPECIAL CASE FOR VECTOR 0
        51   7F 8F    9A  01D6   1553          MOVZBL  #<NUMUBAVEC-1>,R1         ; REST OF VECTORS
           80   53    D0  01DA   1554 30$:     MOVL    R3,(R0)+                  ; FILL VECTOR WITH UNEXP INT
              FA 51   F5  01DD   1555          SOBGTR  R1,30$                    ; FILL ALL VECTORS
                          01E0   1556
                          01E0   1558
                          01E0   1559
                          01E0   1601
                          01E0   1602
                          01E0   1604 ;
                          01E0   1605 ; All memory on the QBUS is main memory.  There is no memory analogous
                          01E0   1606 ; to UNIBUS memory.
                          01E0   1607 ;
                          01E0   1608 ; Now locate the memory controllers and build a list of the addresses
                          01E0   1609 ; at which they are located.  This list is used by the memory error logic
                          01E0   1610 ; in machine-check.  This information must be determined outside of machine-
                          01E0   1611 ; check, since the machine-check code cannot cause another machine-check
                          01E0   1612 ; without causing a cpu double-error halt.
                          01E0   1613 ;
                          01E0   1614 ; The list is a count of controllers, followed by the virtual addresses
                          01E0   1615 ; that are the memory controller CSRs.  Each MSV-11P has a single word CSR.
                          01E0   1616 ;
                          01E0   1617          .ENABLE LSB
  53   00000000'GF    D0  01E0   1618          MOVL    G^EXE$GL_SCB,R3           ; Get SCB address.
              04 A3   DD  01E7   1619          PUSHL   4(R3)                     ; Save current mcheck handler address.
           50   5E    D0  01EA   1620          MOVL    SP,R0                     ; Mark current stack position.
     04 A3   20'AF    DE  01ED   1621          MOVAL   B^MCHK_HANDLER,4(R3)      ; Connect temp mcheck handler.
                          01F2   1622
  51   00000000'GF    D0  01F2   1623          MOVL    G^MMG$GL_SBICONF,R1       ; Get address of SBICONF array.
           51   61    D0  01F9   1624          MOVL    (R1),R1                   ; Get VA of Qbus I/O space.
  51   00002440 8F    C0  01FC   1625          ADDL    #<^O12100+^X1000>,R1      ; Offset to memory controller CSR(772100).
  54   00000000'GF    DE  0203   1626          MOVAL   G^EXE$AL_MEMCSRS,R4       ; Get address of memory CSR count.
        56   04 A4    DE  020A   1627          MOVAL   4(R4),R6                  ; Get address of buffer for CSRs.
                55    D4  020E   1628          CLRL    R5                        ; Initialize index.
                          0210   1629
              6145    B5  0210   1630 50$:     TSTW    (R1)[R5]                  ; Touch possible memory CSR.
                64    D6  0213   1631          INCL    (R4)                      ; Count number of error bits set.
         86   6145    3E  0215   1632          MOVAW   (R1)[R5],(R6)+            ; Save address of this CSR
        F3 55   10    F2  0219   1633 60$:     AOBLSS  #16,R5,50$                ; Loop through all possible CSRs.
                09    11  021D   1634          BRB     70$                       ; Continue with common code.
                          021F   1635 ;
                          021F   1636 ; TEMPORARY MACHINE CHECK HANDLER
                          021F   1637 ;
                          021F   1638
                          021F   1639          .ALIGN  LONG                      ; Align machine-check vector.
                          0220   1640 MCHK_HANDLER:                              ;
                          0220   1641
           26   0F    DA  0220   1642          MTPR    #^XF,#PRUV1$_MCESR        ; Clear machine-check state.
           5E   50    D0  0223   1643          MOVL    R0,SP                     ; Clean mcheck frame from stack.
                F1    11  0226   1644          BRB     60$                       ; Continue looking for memory CSRs.
                          0228   1645
           04 A3 8ED0     0228   1646 70$:     POPL    4(R3)                     ; Restore mcheck handler address.
                          022C   1647
```

```
                           022C  1648            .DISABLE LSB
           56   62    D0   022C  1661            MOVL    ADP$L_CSR(R2),R6         ; Pick up adapter pointer
                51    D4   022F  1662            CLRL    R1                       ; Zero out number of UMR to disable
      0256 C2   51    B0   0231  1686            MOVW    R1,ADP$W_UMR_DIS(R2)     ; Record number disabled
                           0236  1700  ;
                           0236  1701  ; Initialize fields for the Qbus map register allocation.  Since there
                           0236  1702  ; are no map registers for the Micro-VAX I Qbus, initialize the data structures
                           0236  1703  ; so that the standard allocate routine will just return an error.
                           0236  1704  ;
        5C A2   01    D0   0236  1705            MOVL    #1,ADP$L_MRACTMDRS(R2)   ; 1 active map descriptor
64 A2 01F0 8F   51    A3   023A  1706            SUBW3   R1,#496,ADP$W_MRNREGARY(R2);  for a range of 496 registers
                           0241  1707  ;         CLRL    ADP$L_MRACTMDRS(R2)      ; No active descripters.
                           0241  1708  ;         CLRL    ADP$W_MRNREGARY(R2)      ; No registers to allocate,
      015E C2   51    B0   0241  1710            MOVW    R1,ADP$W_MRFREGARY(R2)   ;  starting at register zero.
        62 A2   01    AE   0246  1711            MNEGW   #1,ADP$W_MRNFENCE(R2)    ; Also init "fences" which preceed
      015C C2   01    AE   024A  1712            MNEGW   #1,ADP$W_MRFFENCE(R2)    ;  the two descriptor arrays.
                           024F  1713  ;
                           024F  1714  ; Initialize adapter hardware.
                           024F  1715  ;
           54   62    D0   024F  1716            MOVL    ADP$L_CSR(R2),R4         ; Get CSR address to init
              FDAB'   30   0252  1717            BSBW    UBA$INITIAL              ; And initialize adapter
         01FF 8F      BA   0255  1718            POPR    #^M<R0,R1,R2,R3,R4,R5,R6,R7,R8> ; Restore registers
                      05   0259  1719            RSB                              ; Return
                           025A  1720
                           025A  1728
```

```
                                  025A  1815          .SBTTL  INI$MBADP - BUILD ADP AND INITIALIZE MBA
                                  025A  1816          .SBTTL  INI$DRADP - BUILD ADP AND INITIALIZE DR32
                                  025A  1817          .SBTTL  INI$CIADP - BUILD ADP AND INITIALIZE CI
                                  025A  1818  ;+
                                  025A  1819  ; INI$MBADP IS CALLED AFTER MAPPING THE REGISTERS FOR A MASSBUS ADAPTER.
                                  025A  1820  ; AN ADAPTER CONTROL BLOCK IS ALLOCATED AND FILLED.  A CRB AND IDB ARE
                                  025A  1821  ; ALSO ALLOCATED AND INITIALIZED. THE ADAPTER HARDWARE IS THEN INITIALIZED
                                  025A  1822  ; BY CALLING MBA$INITIAL.
                                  025A  1823  ;
                                  025A  1824  ; INI$DRADP IS CALLED AFTER MAPPING THE REGISTERS FOR THE DR32
                                  025A  1825  ; ADAPTER.  THE ADAPTER CONTROL BLOCK, CRB, AND IDB ARE ALLOCATED
                                  025A  1826  ; AND INITIALIZED.  THE ADAPTER HARDWARE IS THEN INITIALIZED BY
                                  025A  1827  ; CALLING DR$INITIAL.
                                  025A  1828  ;
                                  025A  1829  ; INI$MBADP AND INI$DRADP SHARE COMMON CODE AFTER THE TABLE OF ADAPTER
                                  025A  1830  ; SPECIFIC CONSTANTS IS SELECTED AND STORED IN R8.
                                  025A  1831  ;
                                  025A  1832  ; INPUT:
                                  025A  1833  ;       R4 - nexus identification number of this adapter
                                  025A  1834  ;       R11- offset from beginning of SCB to correct SCB page for this adapter
                                  025A  1835  ;
                                  025A  1836  ; OUTPUTS:
                                  025A  1837  ;       ALL REGISTERS PRESERVED
                                  025A  1838  ;-
                                  025A  1839
               00000000'GF   17   025A  1840  ALONPAGD:JMP    G^INI$ALONONPAGED
                                  0260  1841
                                  0260  1842          .ENABL  LSB
                                  0260  1843
                                  0260  1844  INI$DRADP:                              ; INITIALIZE DR32 DATA STRUCTURES
                                  0260  1845
                                  0260  1855
                                  0260  1856  INI$CIADP:                              ; INITIALIZE CI DATA STRUCTURES
                                  0260  1857
                                  0260  1867
                                  0260  1868  INI$MBADP:                              ; INIT MBA DATA STRUCTURES
                                  0260  1869
```

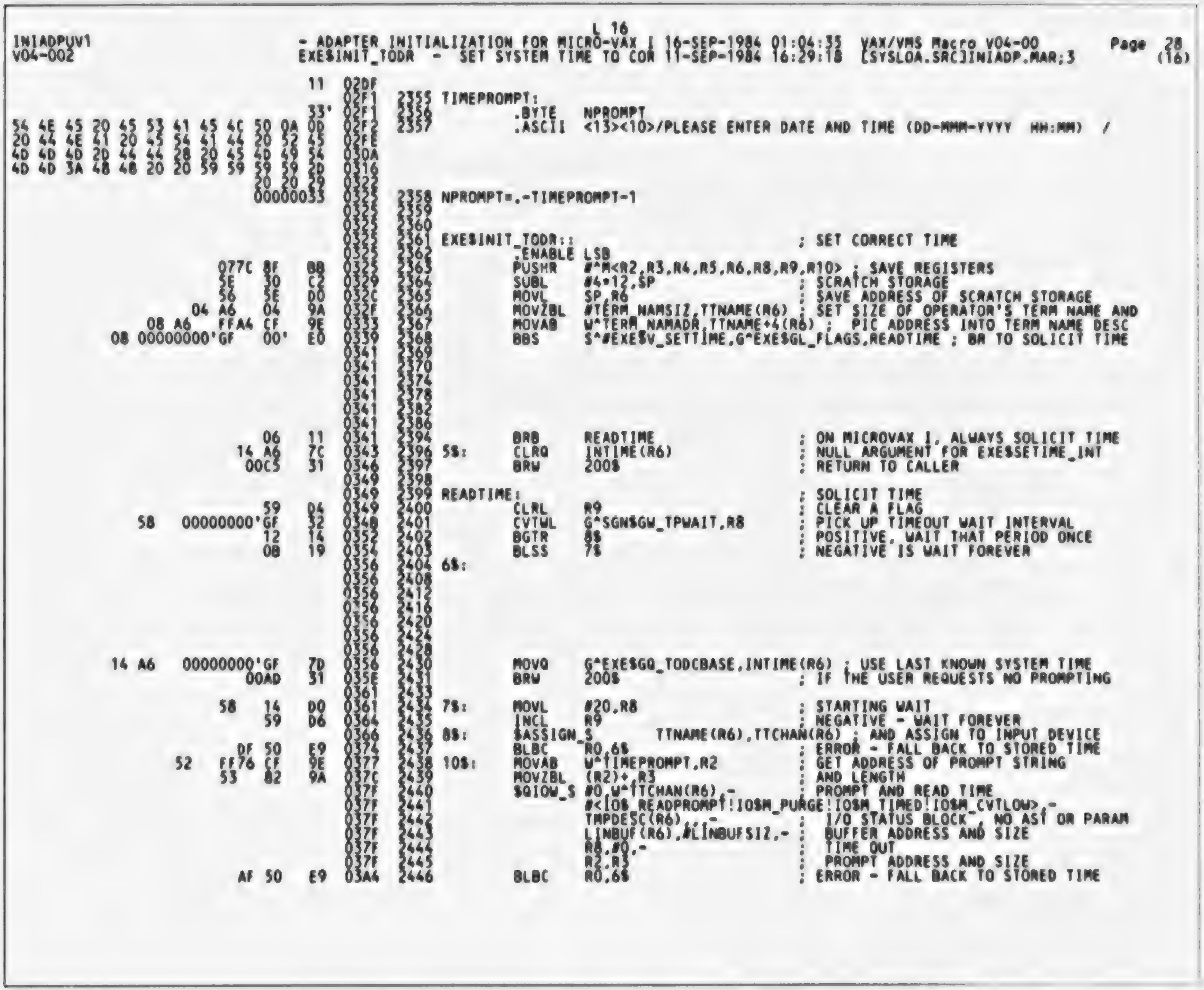
```
        0260         .SBTTL  INI$KDZ11
        0260 ;++
        0260 ;
        0260 ; INPUTS:
        0260 ;           R2 - VA of next free system page
        0260 ;           R3 - VA of system page table entry to be used to map VA in R2
        0260 ;           R4 - nexus identification number of this adapter
        0260 ;
        0260 ; OUTPUTS:
        0260 ;
        0260 ;--
        0260
        0260 INI$KDZ11:
        0260
    05  0260             RSB                                   ; Return to caller.
```

```
                                0261             .SBTTL  INI$CONSOLE, init data structures for console
                                0261      ;++
                                0261      ; FUNCTIONAL DESCRIPTION:
                                0261      ;
                                0261      ;       This routine is executed only once, during system initialization.
                                0261      ;       It initializes the CRB and IDB for boot/console device.
                                0261      ;
                                0261      ;       This routine is called from INIT.
                                0261      ;
                                0261      ; INPUTS:
                                0261      ;
                                0261      ;       R3 -->  DISK [CLASS] DRIVER DDB
                                0261      ;       R4 -->  DISK [CLASS] DRIVER DPT
                                0261      ;       R5 -->  DISK [CLASS] DRIVER UCB
                                0261      ;       R6 -->  RPB
                                0261      ;       R7 -->  ADP FOR EITHER A REAL DISK OR A PORT
                                0261      ;       R9 -->  PORT DRIVER DPT (IF PRESENT)
                                0261      ;       R10-->  PORT DIRVER UCB (IF PRESENT)
                                0261      ;
                                0261      ;--
                                0261
                                0261      INI$CONSOLE::
                                0261             .ENABL  LSB
                                0261
                                0261
                                0261
                                0261
                                0261      ;
                                0261      ; NOW BUILD THE AUXILIARY DATA BLOCKS (CRB,IDB)
                                0261      ;
                                0261      BLD_CRB:
          58   10 A7     D0     0261             MOVL    ADP$L_CRB(R7),R8           ; GET ADDRESS OF CRB IF IT EXISTS
          0E A7   01     B1     0265             CMPW    #AT$_UBA,ADP$W_ADPTYPE(R7); IS THIS A UNIBUS ADAPTER?
                  03     13     0269             BEQL    FILL_CRB                   ; YES, ALLOCATE CRB
                005D     31     026B             BRW     100$                       ; NO, CRB/IDB ALREADY ALLOCATED
                                026E
                                026E      FILL_CRB:
          00000000'9F    16     026E             JSB     @#INI$ALLOC_CRB            ; GO ALLOCATE AND SETUP CRB
  24 A2   9F163FBB 8F    D0     0274             MOVL    #^X9F163FBB,CRB$L_INTD(R2) ; SET PUSHR #^M<R0,...R5>
                                027C                                                ;  JSB @#0 INTO INTERRUPT DISPATCH
          38 A2   57     D0     027C             MOVL    R7,CRB$L_INTD+VEC$L_ADP(R2)      ; SET POINTER TO ADP
             58   52     D0     0280             MOVL    R2,R8                      ; SAVE CRB POINTER
       51   0058 8F      3C     0283             MOVZWL  #<IDB$C_LENGTH+<8*4>>,R1; SIZE TO ALLOCATE FOR IDB
          00000000'9F    16     0288             JSB     @#INI$ALONONPAGED          ; ALLOCATE IDB
          08 A2   51     B0     028E             MOVW    R1,IDB$W_SIZE(R2)          ; SET SIZE OF IDB
          0A A2   09     90     0292             MOVB    #DYN$C_IDB,IDB$B_TYPE(R2); AND STRUCTURE TYPE CODE
          2C A8   52     D0     0296             MOVL    R2,CRB$L_INTD+VEC$L_IDB(R8) ; SET IDB INTO CRB
                                029A
                                029A
          62     58 A6   D0     029A      10$:   MOVL    RPB$L_CSRVIR(R6), -        ; SAVE BOOT DEVICE CSR ADDRESS
                                029E                     IDB$L_CSR(R2)              ; IN INTERRUPT DISPATCH BLOCK
                  11     91     029E             CMPB    #BTD$K_UDA,-               ; LOW ORDER BYTE OF ORIGINAL R0 TELLS
                 66 A6          02A0                     RPB$B_DEVTYP(R6)           ;  BOOT DEVICE TYPE.
                  08     12     02A2             BNEQ    20$                        ; IF NOT BOOTING FROM A UDA BRANCH
                                02A4                                                ;  AROUND.
 00000000'9F     58 A6   D0     02A4             MOVL    RPB$L_CSRVIR(R6), -        ; COPY VIRTUAL ADDRESS OF UDA PORT CSR
                                02AC                     @#BOO$GB_SYSTEMID          ;  TO LOW ORDER LONGWORD OF SYSTEMID
```

```
                               02AC   2122 20$:
           14 A2    57    D0   02AC   2123           MOVL     R7,IDB$L_ADP(R2)           ; POINT IDB TO ADP
           50    1E A6    3C   02B0   2124           MOVZWL   RPB$W_ROOBVEC(R6),R0       ; GET USER SPECIFIED VECTOR
                    0A    12   02B4   2125           BNEQ     30$                        ; BRANCH IF VECTOR SPECIFIED
           50    66 A6    9A   02B6   2126           MOVZBL   RPB$B_DEVTYP(R6),R0        ; ELSE GET DEVICE TYPE CODE
        50   FFFE'CF40    3C   02BA   2127           MOVZWL   W^BOOTVECTOR-2[R0],R0      ; GET DEFAULT INTERRUPT VECTOR
          50    10 B740   9E   02C0   2128 30$:      MOVAB    @ADP$L_VECTOR(R7)[R0],R0; COMPUTE ADDRESS OF VECTOR
           60    26 A8    9E   02C5   2129           MOVAB    CRB$L_INTD+2(R8),(R0)      ; SET ADDR OF INTERRUPT VECTOR
                               02C9   2130
                    60    D7   02C9   2133           DECL     (R0)                       ;  BACK TWO BYTES TO PUSHR, +1 TO
                               02CB   2136                                              ;
                               02CB   2137 100$:
                          05   02CB   2138           RSB                                 ; RETURN
                               02CC   2139           .DISABLE LSB
```

```
                              02CC         .SBTTL  EXE$INI_TIMWAIT - COMPUTE CORRECT TIMEWAIT LOOP VALUES
                              02CC   ;++
                              02CC   ; FUNCTIONAL DESCRIPTION:
                              02CC   ;
                              02CC   ; EXE$INI_TIMWAIT initializes EXE$GL_TENUSEC and EXE$GL_UBDELAY, cells used
                              02CC   ; in the time-wait macros.  The first data cell, EXE$GL_TENUSEC, is the number
                              02CC   ; of times the following loop will be executed in ten u-seconds.  This is
                              02CC   ; done once here to calibrate the loop instead of reading the processor clock.
                              02CC   ; The resulting number is used in the system macros TIMEWAIT and TIMEDWAIT.
                              02CC   ;
                              02CC   ; The first step is to initialize EXE$GL_UBDELAY.  If the bit test instruction
                              02CC   ; in the TIMEWAIT macro is executed too rapidly in a loop, it can saturate the
                              02CC   ; Unibus.  EXE$GL_UBDELAY is used to introduce a 3 microsecond delay loop into
                              02CC   ; the TIMEWAIT bit test loop.
                              02CC   ;
                              02CC   ; This routine is called only once, from INIT.
                              02CC   ;
                              02CC   ; INPUT PARAMETERS:
                              02CC   ;
                              02CC   ;       NONE
                              02CC   ;
                              02CC   ; IMPLICIT INPUTS:
                              02CC   ;
                              02CC   ;       Time-of-day processor clock.
                              02CC   ;       Interval timers.
                              02CC   ;
                              02CC   ; OUTPUT PARAMETERS:
                              02CC   ;
                              02CC   ;       R0 - Destroyed.
                              02CC   ;
                              02CC   ; IMPLICIT OUTPUTS:
                              02CC   ;
                              02CC   ;       EXE$GL_TENUSEC - set to appropriate value to make TIMEWAIT and TIMEDWAIT
                              02CC   ;                        macros loop for 10 micro-seconds.
                              02CC   ;
                              02CC   ;       EXE$GL_UBDELAY - set to appropriate value to make TIMEWAIT and TIMEDWAIT
                              02CC   ;                        macros loop for 3 micro-seconds in the unibus delay
                              02CC   ;                        loop.
                              02CC   ;
                              02CC   ;--
                              02CC
                              02CC EXE$INI_TIMWAIT::                        ; Initialize time-wait data cells
    00000000'GF    01    9A   02CC         MOVZBL  #1,G^EXE$GL_UBDELAY      ; Set UV1 value same as 11/730
    00000000'GF    01    9A   02D3         MOVZBL  #1,G^EXE$GL_TENUSEC      ; Set UV1 value same as 11/73
                         05   02DA         RSB                              ; Return
```

```
                                     02DB         .SBTTL  EXE$INIT_TODR  -  SET SYSTEM TIME TO CORRECT VALUE AT STARTUP
                                     02DB  ;++
                                     02DB  ; FUNCTIONAL DESCRIPTION:
                                     02DB  ;
                                     02DB  ;       EXE$INIT_TODR SOLICITS THE CORRECT TIME FROM THE OPERATOR IF NECESSARY,
                                     02DB  ;       CONVERTS THE ASCII RESPONSE TO BINARY FORMAT AND CALLS AN INTERNAL
                                     02DB  ;       ENTRY POINT OF THE $SETIME SYSTEM SERVICE TO SET THE NEW SYSTEM TIME
                                     02DB  ;       IN MEMORY WITHOUT MODIFYING THE CONTENTS OF THE SYSTEM DISK.
                                     02DB  ;
                                     02DB  ;       IF THE TIME WOULD NORMALLY BE SOLICITED FROM AN OPERATOR, BECAUSE
                                     02DB  ;       THE HARDWARE TIME OF YEAR CLOCK IS ZERO, THEN THE SYSGEN PARAMETER
                                     02DB  ;       "TPWAIT" IS CHECKED.  IF IT IS ZERO, THEN IT IS ASSUMED THAT NO
                                     02DB  ;       OPERATOR IS PRESENT AND THE SYSTEM IS BOOTED USING THE LAST TIME
                                     02DB  ;       RECORDED IN THE SYSTEM IMAGE.  IF THE PARAMETER IS NON ZERO THEN
                                     02DB  ;       THAT TIME IS USED AS THE MAXIMUM TIME TO WAIT BEFOR ASSUMING THAT
                                     02DB  ;       THERE IS NO OPERATOR AND BOOTING ANY WAY.  IF THE PARAMETER IS
                                     02DB  ;       NEGATIVE, THE SYSTEM WILL WAIT FOREVER.
                                     02DB  ;
                                     02DB  ;       THIS ROUTINE IS CALLED ONLY ONCE, FROM SYSINIT OR STASYSGEN.
                                     02DB  ;
                                     02DB  ; INPUT PARAMETERS:
                                     02DB  ;
                                     02DB  ;       NONE
                                     02DB  ;
                                     02DB  ; IMPLICIT INPUTS:
                                     02DB  ;
                                     02DB  ;       TIME-OF-DAY PROCESSOR CLOCK.
                                     02DB  ;
                                     02DB  ; OUTPUT PARAMETERS:
                                     02DB  ;
                                     02DB  ;       R0,R1 - DESTROYED
                                     02DB  ;
                                     02DB  ; IMPLICIT OUTPUTS:
                                     02DB  ;
                                     02DB  ;       EXE$GQ_SYSTIME - SET TO CURRENT TIME IN 100 NANOSECOND UNITS SINCE
                                     02DB  ;                        17-NOV-1858  00:00:00.
                                     02DB  ;
                                     02DB  ;--
                                     02DB
                                     02DB  ;
                                     02DB  ; Stack storage offsets:
                                     02DB  ;
                          00000000   02DB  TTCHAN  = ^X00                        ; CHANNEL FOR TERMINAL (LONGWORD)
                          00000004   02DB  TTNAME  = ^X04                        ; STRING DESCRIPTOR FOR OPERATOR'S TERM
                          0000000C   02DB  TMPDESC = ^X0C                        ; TEMPORY STRING DESCRIPTOR (QUADWORD)
                          00000014   02DB  INTIME  = ^X14                        ; INPUT TIME VALUE (QUADWORD)
                          0000001C   02DB  LINBUF  = ^X1C                        ; INPUT LINE BUFFER (5 LONGWORDS)
                          00000014   02DB  LINBUFSIZ = ^X14                      ;  (LENGTH OF LINE BUFFER IN BYTES)
                                     02DB
                                     02DB  ;
                                     02DB  ; PURE DATA
                                     02DB  ;
                                     02DB  TERM_NAMADR:
                       30 41 50 4F   02DB          .ASCII  \OPA0\                ; DEVICE NAME FOR OPERATOR'S TERMINAL
                          00000004   02DF  TERM_NAMSIZ = . - TERM_NAMADR
74 61 64 20 64 69 6C 61 76 6E 69 00' 02DF  TIMERR: .ASCIC  \invalid date/time\     ;
                 65 6D 69 74 2F 65   02EB
```



```
                                                                 L 16
INIADPUV1                            - ADAPTER INITIALIZATION FOR MICRO-VAX I 16-SEP-1984 01:04:35  VAX/VMS Macro V04-00        Page  28
V04-002                              EXE$INIT_TODR  -  SET SYSTEM TIME TO COR 11-SEP-1984 16:29:18  [SYSLOA.SRC]INIADP.MAR;3        (16)

                                11  02DF
                                    02F1  2355 TIMEPROMPT:
                               33'  02F1  2356         .BYTE   NPROMPT
54 4E 45 20 45 53 41 45 4C 50 0A 0D  02F2  2357         .ASCII  <13><10>/PLEASE ENTER DATE AND TIME (DD-MMM-YYYY  HH:MM)  /
20 44 4E 41 20 45 54 41 44 20 52 45  02FE
4D 4D 4D 2D 44 44 28 20 45 4D 49 54  030A
4D 4D 3A 48 48 20 20 59 59 59 59 2D  0316
                          20 20 29  0322
                          00000033  0325  2358 NPROMPT=.-TIMEPROMPT-1
                                    0325  2359
                                    0325  2360
                                    0325  2361 EXE$INIT_TODR::                              ; SET CORRECT TIME
                                    0325  2362         .ENABLE LSB
                  077C 8F      BB   0325  2363         PUSHR   #^M<R2,R3,R4,R5,R6,R8,R9,R10> ; SAVE REGISTERS
                     5E 30     C2   0329  2364         SUBL    #4*12,SP                     ; SCRATCH STORAGE
                     56 5E     D0   032C  2365         MOVL    SP,R6                        ; SAVE ADDRESS OF SCRATCH STORAGE
                  04 A6 04     9A   032F  2366         MOVZBL  #TERM_NAMSIZ,TTNAME(R6)      ; SET SIZE OF OPERATOR'S TERM NAME AND
             08 A6  FFA4 CF    9E   0333  2367         MOVAB   W^TERM_NAMADR,TTNAME+4(R6)   ;  PIC ADDRESS INTO TERM NAME DESC
      08 00000000'GF   00'     E0   0339  2368         BBS     S^#EXE$V_SETTIME,G^EXE$GL_FLAGS,READTIME ; BR TO SOLICIT TIME
                                    0341  2369
                                    0341  2370
                                    0341  2374
                                    0341  2378
                                    0341  2382
                                    0341  2386
                        06     11   0341  2394         BRB     READTIME                     ; ON MICROVAX I, ALWAYS SOLICIT TIME
                     14 A6     7C   0343  2396 5$:     CLRQ    INTIME(R6)                   ; NULL ARGUMENT FOR EXE$SETIME_INT
                      00C5     31   0346  2397         BRW     200$                         ; RETURN TO CALLER
                                    0349  2398
                                    0349  2399 READTIME:                                    ; SOLICIT TIME
                        59     D4   0349  2400         CLRL    R9                           ; CLEAR A FLAG
        58    00000000'GF      32   034B  2401         CVTWL   G^SGN$GW_TPWAIT,R8           ; PICK UP TIMEOUT WAIT INTERVAL
                        12     14   0352  2402         BGTR    8$                           ; POSITIVE, WAIT THAT PERIOD ONCE
                        08     19   0354  2403         BLSS    7$                           ; NEGATIVE IS WAIT FOREVER
                                    0356  2404 6$:
                                    0356  2408
                                    0356  2412
                                    0356  2416
                                    0356  2420
                                    0356  2424
                                    0356  2428
     14 A6    00000000'GF      7D   0356  2430         MOVQ    G^EXE$GQ_TODCBASE,INTIME(R6) ; USE LAST KNOWN SYSTEM TIME
                      00AD     31   035E  2431         BRW     200$                         ; IF THE USER REQUESTS NO PROMPTING
                                    0361  2433
                   58   14     D0   0361  2434 7$:     MOVL    #20,R8                       ; STARTING WAIT
                        59     D6   0364  2435         INCL    R9                           ; NEGATIVE - WAIT FOREVER
                                    0366  2436 8$:     $ASSIGN_S      TTNAME(R6),TTCHAN(R6) ; AND ASSIGN TO INPUT DEVICE
                     DF 50     E9   0374  2437         BLBC    R0,6$                        ; ERROR - FALL BACK TO STORED TIME
              52   FF76 CF     9E   0377  2438 10$:    MOVAB   W^TIMEPROMPT,R2              ; GET ADDRESS OF PROMPT STRING
                   53   82     9A   037C  2439         MOVZBL  (R2)+,R3                     ; AND LENGTH
                                    037F  2440         $QIOW_S #0,W^TTCHAN(R6),-            ; PROMPT AND READ TIME
                                    037F  2441                 #<IO$_READPROMPT!IO$M_PURGE!IO$M_TIMED!IO$M_CVTLOW>,-
                                    037F  2442                 TMPDESC(R6),,-               ;  I/O STATUS BLOCK , NO AST OR PARAM
                                    037F  2443                 LINBUF(R6),#LINBUFSIZ,-      ;  BUFFER ADDRESS AND SIZE
                                    037F  2444                 R8,#0,-                      ;  TIME OUT
                                    037F  2445                 R2,R3                        ;  PROMPT ADDRESS AND SIZE
                     AF 50     E9   03A4  2446         BLBC    R0,6$                        ; ERROR - FALL BACK TO STORED TIME
```

```
            54     0C  A6    7D  03A7  2447          MOVQ     TMPDESC(R6),R4             ; GET COMPLETION STATUS
                   0D  54    E8  03AB  2448          BLBS     R4,20$                     ; CONTINUE IF SUCCESSFUL READ
                   A5  59    E9  03AE  2449          BLBC     R9,6$                      ; FAILED ON ONE-TIME READ, RETURN
            58     01 A848   9E  03B1  2450          MOVAB    1(R8)[R8],P8               ; (2 * TIMEOUT) + 1
                   58  58    3C  03B6  2451          MOVZWL   R8,R8                      ; BOUND TIMEOUT
                       BC    11  03B9  2452          BRB      10$                        ; TRY AGAIN FOR TIME
                                 03BB  2453 20$:                                         ; SOMETHING WAS INPUT
         0C A6     0E  A6    3C  03BB  2454          MOVZWL   TMPDESC+2(R6),TMPDESC(R6)  ; FORM DESCRIPTOR FOR BUFFER
         10 A6     1C  A6    9E  03C0  2455          MOVAB    LINBUF(R6),TMPDESC+4(R6)   ; SET DESCRIPTOR ADDRESS
                                 03C5  2456          $BINTIM_S TMPDESC(R6),INTIME(R6)    ; CONVERT TO BINARY TIME
                   05  50    E9  03D2  2457          BLBC     R0,89$                     ; INVALID TIME
                   1B  A6    D5  03D5  2458          TSTL     INTIME+4(R6)               ; CHECK FOR DELTA TIME
                       2A    14  03D8  2459          BGTR     100$                       ; BRANCH IF NOT - OK
                                 03DA  2460 89$:                                         ; INVALID TIME VALUE INPUT
            52     FF01 CF   9E  03DA  2461          MOVAB    W^TIMERR,R2                ; ADDRESS OF ERROR MESSAGE
                   53  82    9A  03DF  2462          MOVZBL   (R2)+,R3                   ; GET STRING LENGTH
                                 03E2  2463          $QIOW_S  #0,TTCHAN(R6),-            ; GIVE ERROR MESSAGE
                                 03E2  2464                   #IO$_WRITEVBLK,-           ;
                                 03E2  2465                   ,,,-                       ;  NO I/O STATUS,AST OR AST PARAM
                                 03E2  2466                   (R2),R3 ,-                 ;  BUFFER ADDRESS, LENGTH
                                 03E2  2467                   #0,#32                     ;  SET CARRIAGE CONTROL TO CR/LF
                     FF73    31  0401  2468          BRW      10$                        ; AND TRY AGAIN
                                 0404  2469 100$:                                        ; EXIT
                                 0404  2470          $DASSGN_S TTCHAN(R6)                ; DE-ASSIGN TERMINAL CHANNEL
                   14  A6    7F  040E  2471 200$:    PUSHAQ   INTIME(R6)                 ; SET NEW SYSTEM TIME
         00000000'GF   01    FB  0411  2472          CALLS    #1,G^EXE$SETIME_INT        ; USE TODR CLOCK TO SET SYSTEM TIME
00000000'GF   00000000'GF    7D  0418  2473          MOVQ     G^EXE$GQ_TODCBASE,G^EXE$GQ_BOOTTIME ; SAVE BOOT TIME
                   5E  30    C0  0423  2474          ADDL     #12*4,SP                   ; CLEAN OFF SCRATCH STORAGE
                 077C  8F    BA  0426  2475          POPR     #^M<R2,R3,R4,R5,R6,R8,R9,R10> ; RESTORE REGISTERS
                                 042A  2476
                                 042A  2477 ;
                                 042A  2478 ; Fall through into the deallocate logic.
                                 042A  2479 ;
                                 042A  2480 ;        RSB                                 ; *** This goes in if another piece of
                                 042A  2481                                              ; *** initialization code is added that
                                 042A  2482                                              ; *** is executed after EXE$INI_TIMWAIT.
                                 042A  2483          .DISABLE LSB
                                 042A  2484
```

```
                                  042A  DEAL_INIT_CODE:                           ; DEALLOCATE THE INITIALIZATION CODE
                                  042A  ;
                                  042A  ; It is the duty of the last-executed, loadable initialization
                                  042A  ; routine to make itself and all other such routines disappear, i.e.,
                                  042A  ; release the space they occupy to non-paged pool.  Each routine's vector
                                  042A  ; must be disconnected, e.g., be made to point to the symbol, EXE$LOAD_ERROR.
                                  042A  ;
                                  042A  ; NOTE:  This means that new initialization routines should be added
                                  042A  ;        to this module in a particular order, not necessarily at the
                                  042A  ;        end of the module!
                                  042A  ;
                                  042A          .ENABLE LSB
                 7E     52     7D 042A          MOVQ    R2,-(SP)                  ; Save some registers
                                  042D
                                  042D  ;
                                  042D  ; First find the vectors that point to these initialization routines
                                  042D  ; and reset them to point to EXE$LOAD_ERROR.
                                  042D  ;
             50    0000'CF     9E 042D          MOVAB   W^SYSL$BEGIN,R0           ; Compute bounds of releasable piece:
     51    50  00000000'8F     C1 0432          ADDL3   #<STAY_HEADER-SYSL$BEGIN>,R0,R1 ; starting and ending addresses.
           52  00000000'GF     9E 043A          MOVAB   G^EXE$AL_LOAVEC,R2        ; Get starting address of vectors.
           53  00000000'GF     9E 0441          MOVAB   G^EXE$LOAD_ERROR,R3       ; Get end of vectors.
            9F17 8F     62     B1 0448  10$:    CMPW    (R2),#^X9FT7              ; Is this JMP @# ?
                        1B     13 044D          BEQL    30$                       ; Br if yes, skip past it.
           80 8F     03 A2     91 044F          CMPB    3(R2),#^X80               ; Is this a system space address
                        16     12 0454          BNEQ    40$                       ; Br if no, assume it's a HALT instr.
                 50     62     D1 0456          CMPL    (R2),R0                   ; Is address before the releasable
                        0C     1F 0459          BLSSU   20$                       ;  piece of memory?  Br on yes.
                 51     62     D1 045B          CMPL    (R2),R1                   ; Is address after the releasable
                        07     1A 045E          BGTRU   20$                       ;  piece of memory?  Br on yes.
        62     00000000'GF     9E 0460          MOVAB   G^EXE$LOAD_ERROR,(R2)     ; Reset this vector.
                 52     02     C0 0467  20$:    ADDL    #2,R2                     ; Point past this vector.
                        52     D6 046A  30$:    INCL    R2                        ; Come here to point past JMP @#.
                        52     D6 046C  40$:    INCL    R2                        ; Come here to point past HALT.
                 53     52     D1 046E          CMPL    R2,R3                     ; Past the end of the vectors?
                        D5     1F 0471          BLSSU   10$                       ; Keep searching vectors.
                                  0473  ;
                                  0473  ; Now release the memory to non-paged pool.
                                  0473  ;
             50    0000'CF     9E 0473          MOVAB   W^SYSL$BEGIN,R0           ; Point to start of module
             51    0000'8F     3C 0478          MOVZWL  #<STAY_HEADER-SYSL$BEGIN>,R1 ; Length to vaporize
                     FB8C'     31 047D          BRW     50$                       ; Br to code that is not released.
                                  0480
                            00000000            .PSECT  $$$INIT__END,PAGE         ; 'PAGE' SINCE 16-BYTE ALIGN IS NOT
                                  0000
                                  0000  STAY_HEADER:
             00000000 00000000    0000          .LONG   0,0
                             0000' 0008          .WORD   <SYSL$END-STAY_HEADER>
                               62 000A          .BYTE   DYN$C_LOADCODE
                               00 000B          .BYTE   0
                                  000C
             00000000'9F       16 000C  50$:    JSB     @#EXE$DEANONPGDSIZ        ; Just the smile on the Chesire cat
                 52     8E     7D 0012          MOVQ    (SP)+,R2                  ; Restore
                               05 0015          RSB                               ; Return.
                                  0016
                                  0016          .DISABLE LSB
                                  0016          .END
```

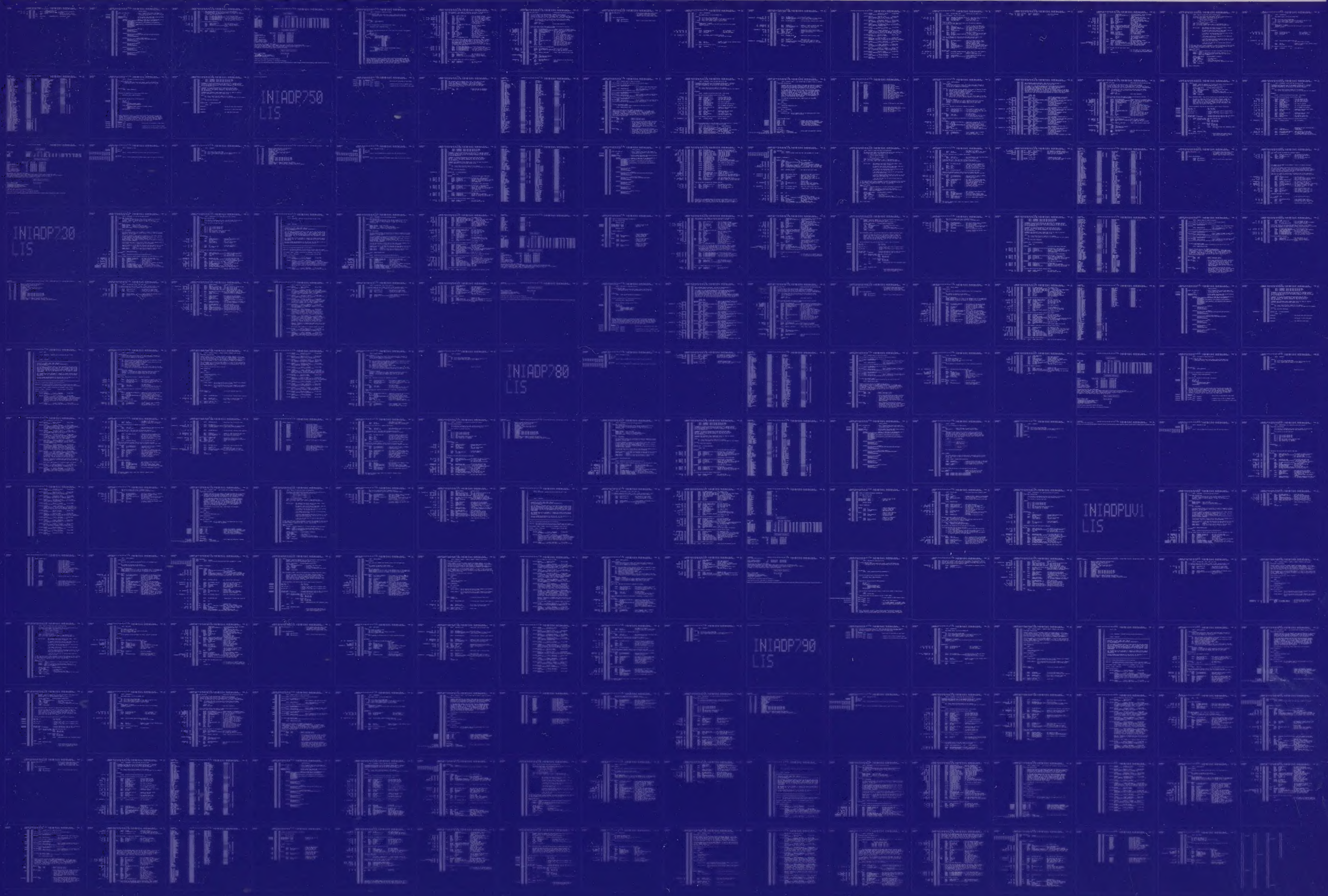
```
$$$VMSDEFINED                  = 00000001                   EXE$DEANONPGDSIZ               ********   X  0A
$$T1                           = 00000001                   EXE$GL_CONFREG                 ********   X  09
ADAPTERS                         00000000 R       02        EXE$GL_CONFREGL                ********   X  09
ADP$B_TYPE                     = 0000000A                   EXE$GL_FLAGS                   ********   X  09
ADP$C_UBAADPLEN                = 00000258                   EXE$GL_NUMNEXUS                ********   X  09
ADP$L_CRB                      = 00000010                   EXE$GL_RPB                     ********   X  09
ADP$L_CSR                      = 00000000                   EXE$GL_SCB                     ********   X  09
ADP$L_DPQFL                    = 00000014                   EXE$GL_TENUSEC                 ********   X  09
ADP$L_LINK                     = 00000004                   EXE$GL_UBDELAY                 ********   X  09
ADP$L_MRACTMDRS                = 0000005C                   EXE$GQ_BOOTTIME                ********   X  09
ADP$L_MRQFL                    = 00000030                   EXE$GQ_TODCBASE                ********   X  09
ADP$L_VECTOR                   = 00000010                   EXE$INIT_TODR                  00000325 RG    09
ADP$W_ADPTYPE                  = 0000000E                   EXE$INI_TIMWAIT                000002CC RG    09
ADP$W_DPBITMAP                 = 00000060                   EXE$LOAD_ERROR                 ********   X  09
ADP$W_MRFFENCE                 = 0000015C                   EXE$OUTZSTRING                 ********   X  09
ADP$W_MRFREGARY                = 0000015E                   EXE$SETIME_INT                 ********   X  09
ADP$W_MRNFENCE                 = 00000062                   EXE$V_SETTIME                  ********   X  09
ADP$W_MRNREGARY                = 00000064                   FILL_CRB                       0000026E R     09
ADP$W_SIZE                     = 00000008                   GET_GEN_TYPE                   0000006B R     09
ADP$W_TR                       = 0000000C                   GET_TYPE                       00000068 R     09
ADP$W_UMR_DIS                  = 00000256                   IDB$B_TYPE                   = 0000000A
ADPLINK                          ********   X     09        IDB$C_LENGTH                 = 00000038
ALONPAGD                         0000025A R       09        IDB$L_ADP                    = 00000014
AT$_UBA                        = 00000001                   IDB$L_CSR                    = 00000000
BI_BUS_CODE                    = 80000000                   IDB$W_SIZE                   = 00000008
BI_CPU                         = 00000000                   INI$ALLOC_CRB                  ********   X  09
BI_CSR_LEN                     = 00000002                   INI$ALONONPAGED                ********   X  09
BI_LIKE                        = 00000000                   INI$CIADP                      00000260 R     09
BLD_CRB                          00000261 R       09        INI$CONSOLE                    00000261 RG    09
BOO$GB_SYSTEMID                  ********   X     09        INI$DRADP                      00000260 R     09
BOO$GL_SPTFREH                   ********   X     09        INI$IOMAP                      00000000 RG    09
BOO$GL_SPTFREL                   ********   X     09        INI$KDZ11                      00000260 R     09
BOOTVECTOR                       00000000 R       08        INI$MBADP                      00000260 R     09
BTD$K_UDA                      = 00000011                   INI$MPMADP                     ********   X  06
BUS_CODE_OFFSET                = FFFFFFFC                   INI$UBADP                      00000174 R     09
BUS_CSR_LEN                      00000004 R       08        INI$UBSPACE                    00000157 R     09
CONFIG_IOSPACE                   00000059 R       09        INIT_ROUTINES                  00000000 R     06
CONFREG                          00000020 R       08        INTIME                       = 00000014
CONFREGL                         00000160 R       08        IO$M_CVTLOW                    ********   X  09
CPU_ADPSIZE                      0000000D R       08        IO$M_PURGE                     ********   X  09
CPU_TYPE                       = 00000007                   IO$M_TIMED                     ********   X  09
CR                             = 0000000D                   IO$_READPROMPT                 ********   X  09
CRB$L_INTD                     = 00000024                   IO$_WRITEVBLK                  ********   X  09
CREATE_ARRAYS                    000000CA R       09        IOUV1$AL_QBOSP               = 20000000
CSR_LEN_OFFSET                 = FFFFFFFB                   LF                           = 0000000A
DEAL_INIT_CODE                   0000042A R       09        LINBUF                       = 0000001C
DIRECT_VEC_NODE_CNT              00000009 R       08        LINBUFSIZ                    = 00000014
DYN$C_ADP                      = 00000001                   MAP_NEXUS                      000000A6 R     09
DYN$C_CONF                     = 00000007                   MAP_PAGES                      00000123 R     09
DYN$C_IDB                      = 00000009                   MAXNEXUS                     = 00000040
DYN$C_INIT                     = 00000063                   MCHK_HANDLER                   00000220 R     09
DYN$C_LOADCODE                 = 00000062                   MMG$GL_SBICONF                 ********   X  09
END_NEXUS                        000000C7 R       09        MMG$GL_SPTBASE                 ********   X  09
ERROR_HALT                       00000149 R       09        NDT$_BUA                     = 80000102
ERROR_HALT_1                     0000014E R       09        NDT$_CI                      = 00000038
EXE$AL_LOAVEC                    ********   X     09        NDT$_DR32                    = 00000030
EXE$AL_MEMCSRS                   ********   X     09        NDT$_KDZ11                   = 80000105
```

```
NDT$_MB                         = 00000020                  SYS$ASSIGN                    ********  GX  09
NDT$_MEM1664NI                  = 00000012                  SYS$BINTIM                    ********  GX  09
NDT$_MEM16I                     = 00000011                  SYS$DASSGN                    ********  GX  09
NDT$_MEM16NI                    = 00000010                  SYS$QIOW                      ********  GX  09
NDT$_MEM256EIL                  = 00000071                  SYSL$BEGIN                    ********   X  09
NDT$_MEM256EIU                  = 00000073                  SYSL$END                      ********   X  0A
NDT$_MEM256I                    = 00000074                  TERM_NAMADR                   000002DB R    09
NDT$_MEM256NIL                  = 00000070                  TERM_NAMSIZ                 = 00000004
NDT$_MEM256NIU                  = 00000072                  TIMEPROMPT                    000002F1 R    09
NDT$_MEM4I                      = 00000009                  TIMERR                        000002DF R    09
NDT$_MEM4NI                     = 00000008                  TMPDESC                     = 0000000C
NDT$_MEM64EIL                   = 00000069                  TTCHAN                      = 00000000
NDT$_MEM64EIU                   = 0000006B                  TTNAME                      = 00000004
NDT$_MEM64I                     = 0000006C                  UBA$INITIAL                   ********   X  09
NDT$_MEM64NIL                   = 00000068                  UBA$INT0                      ********   X  09
NDT$_MEM64NIU                   = 0000006A                  UBA$UNEXINT                   ********   X  09
NDT$_MPM0                       = 00000040                  VA$M_SYSTEM                 = 80000000
NDT$_MPM1                       = 00000041                  VEC$L_ADP                   = 00000014
NDT$_MPM2                       = 00000042                  VEC$L_IDB                   = 00000008
NDT$_MPM3                       = 00000043
NDT$_SCORMEM                    = 80000001
NDT$_UB0                        = 00000028
NDT$_UB1                        = 00000029
NDT$_UB2                        = 0000002A
NDT$_UB3                        = 0000002B
NEXUSDESC                         00000014 R      08
NOSPT                             00000260 R      08
NPROMPT                         = 00000033
NUMUBAVEC                       = 00000080
NUM_PAGES                         00000000 R      04
NXT_NEXUS                         00000065 R      09
PA                              = 20000000
PR$_SID_TYP730                  = 00000003
PR$_SID_TYP750                  = 00000002
PR$_SID_TYP780                  = 00000001
PR$_SID_TYP790                  = 00000004
PR$_SID_TYP8NN                  = 00000006
PR$_SID_TYP8SS                  = 00000005
PR$_SID_TYPUV1                  = 00000007
PRUV1$_MCESR                    = 00000026
PTE$C_KW                        = 10000000
PTE$M_VALID                     = 80000000
READTIME                          00000349 R      09
RPB$B_DEVTYP                    = 00000066
RPB$L_ADPVIR                    = 00000060
RPB$L_BOOTR1                    = 00000020
RPB$L_CSRPHY                    = 00000054
RPB$L_CSRVIR                    = 00000058
RPB$W_ROUBVEC                   = 0000001E
SBICONF                           00000060 R      08
SBI_BUS_CODE                    = 00000000
SBI_CPU                         = 00000000
SBI_CSR_LEN                     = 00000001
SBI_LIKE                        = 00000001
SGN$GW_TPWAIT                     ********    X   09
STAY_HEADER                       00000000 R      0A
SW_BUS_CODE                       00000005 R      08
```

```
                                                  +----------------+
                                                  ! Psect synopsis !
                                                  +----------------+

PSECT name                      Allocation           PSECT No.  Attributes
----------                      ----------           ---------  ----------
.  ABS  .                       00000000  (     0.)  00 (  0.)  NOPIC    USR    CON    ABS    LCL NOSHR NOEXE NORD   NOWRT NOVEC BYTE
$ABS$                           00000000  (     0.)  01 (  1.)  NOPIC    USR    CON    ABS    LCL NOSHR   EXE   RD     WRT NOVEC BYTE
$$$INIT$DATA0                   00000074  (   116.)  02 (  2.)  NOPIC    USR    CON    REL    LCL NOSHR   EXE   RD     WRT NOVEC BYTE
$$$INIT$DATA1                   00000000  (     0.)  03 (  3.)  NOPIC    USR    CON    REL    LCL NOSHR   EXE   RD     WRT NOVEC BYTE
$$$INIT$DATA2                   0000003A  (    58.)  04 (  4.)  NOPIC    USR    CON    REL    LCL NOSHR   EXE   RD     WRT NOVEC BYTE
$$$INIT$DATA3                   00000000  (     0.)  05 (  5.)  NOPIC    USR    CON    REL    LCL NOSHR   EXE   RD     WRT NOVEC BYTE
$$$INIT$DATA4                   00000074  (   116.)  06 (  6.)  NOPIC    USR    CON    REL    LCL NOSHR   EXE   RD     WRT NOVEC BYTE
$$$INIT$DATA5                   00000000  (     0.)  07 (  7.)  NOPIC    USR    CON    REL    LCL NOSHR   EXE   RD     WRT NOVEC BYTE
$$$INIT$DATA                    00000289  (   649.)  08 (  8.)  NOPIC    USR    CON    REL    LCL NOSHR   EXE   RD     WRT NOVEC LONG
$$$INIT$CODE                    00000480  (  1152.)  09 (  9.)  NOPIC    USR    CON    REL    LCL NOSHR   EXE   RD     WRT NOVEC QUAD
$$$INIT__END                    00000016  (    22.)  0A ( 10.)  NOPIC    USR    CON    REL    LCL NOSHR   EXE   RD     WRT NOVEC PAGE


                                               +------------------------+
                                               ! Performance indicators !
                                               +------------------------+

Phase                        Page faults    CPU Time       Elapsed Time
-----                        -----------    --------       ------------
Initialization                        35    00:00:00.06    00:00:01.64
Command processing                   141    00:00:00.45    00:00:03.20
Pass 1                               507    00:00:12.82    00:00:47.78
Symbol table sort                      0    00:00:01.65    00:00:06.88
Pass 2                               234    00:00:03.78    00:00:17.39
Symbol table output                   24    00:00:00.13    00:00:00.82
Psect synopsis output                  4    00:00:00.03    00:00:00.03
Cross-reference output                 0    00:00:00.00    00:00:00.00
Assembler run totals                 947    00:00:18.92    00:01:17.74

The working set limit was 2100 pages.
132724 bytes (260 pages) of virtual memory were used to buffer the intermediate code.
There were 90 pages of symbol table space allocated to hold 1600 non-local and 24 local symbols.
2546 source lines were read in Pass 1, producing 36 object records in Pass 2.
42 pages of virtual memory were used to define 40 macros.


                                              +--------------------------+
                                              ! Macro library statistics !
                                              +--------------------------+

Macro library name                            Macros defined
------------------                            --------------
_$255$DUA28:[SYS.OBJ]LIB.MLB;1                        19
_$255$DUA28:[SYSLIB]STARLET.MLB;2                     14
TOTALS (all libraries)                                33

1745 GETS were required to define 33 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS$:INIADPUV1/OBJ=OBJ$:INIADPUV1 MSRC$:CPUSWUV1/UPDATE=(ENH$:CPUSWUV1)+MSRC$:INIADP/UPDATE=(ENH$:INIADP)+EXECML$/LIB
```

0396 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY



0397 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY